Prototype based Machine Learning for Clinical Proteomics

Dissertation

zur Erlangung des Grades eines Doktors

der Naturwissenschaften

vorgelegt von

Dipl.-Inf. Frank-Michael Schleif

aus Leipzig

genehmigt von der Fakultät für Mathematik/Informatik u. Maschinenbau
der Technischen Universität Clausthal,

Tag der mündlichen Prüfung
08.12.2006

Vorsitzender der Promotionskommission:	Prof. Dr. rer. nat. Michael Demuth
Hauptberichterstatterin:	Prof. Dr. rer. nat. Barbara Hammer Technische Universität Clausthal
Berichterstatter :	PD Dr. rer. nat. habil. Thomas Villmann Universität Leipzig

Abstract

Clinical proteomics opens the way towards new insights into many diseases on a level of detail not available before. One of the most promising measurement techniques supporting this approach is mass spectrometry based clinical proteomics.

The analysis of the high dimensional data obtained from mass spectrometry asks for sophisticated, problem adequate preprocessing and data analysis approaches. Ideally, automatic analysis tools provide insight into their behavior and the ability to extract further information, relevant for an understanding of the clinical data or applications such as biomarker discovery.

Prototype based algorithms constitute efficient, intuitive and powerful machine learning methods which are very well suited to deal with high dimensional data and which allow good insight into their behavior by means of prototypical data locations. They have already successfully been applied to various problems in bioinformatics. The goal of this thesis is to extend prototype based methods, in such a way that they become suitable machine learning tools for typical problems in clinical proteomics.

To achieve better adapted classification borders, tailored to the specific data distributions which occur in clinical proteomics, the prototype based algorithms are extended by local relevance determination and other problem specific metrics.

Fuzzy classification is introduced into prototype approaches to allow for the integration of insecure class label information and to provide the possibility to judge the safety of the classification as it is typically needed in clinical domains. Further margin based active learning is developed to achieve a faster training and better generalization ability ideally suited for the often complex classification problems in clinical research. All algorithms are extensively tested for several clinical data sets in the context of cancer research, including publicly available benchmark data sets as well as recent clinical data sets obtained by state-of-the-art biotechnology.

Keywords: vector quantization, self-organization, local relevance learning,
fuzzy classification, active learning,
clinical proteomics, mass spectrometry

Acknowledgments

Intelligent and efficient data analysis is the main motivation of my work. A successful combination of the sophisticated concepts of pattern recognition methods with the knowledge of human experts often helps dramatically to solve complex problems. Self-organizing methods thereby show promising progress in research and application due to there biological motivation and formal derivation. They point into new directions of future applications and further topics of research. Dr. rer. nat. habil. Thomas Villmann has encouraged me for research in the interesting field of self organized pattern recognition. In countless hours of discussions he contributed to improvements of this work. He was also a delightful research fellow with unconventional new ideas and sometimes enthusiastic and sometimes moderating words on rumbling research ideas.

Prof. Barbara Hammer was and is an ongoing source of inspiration and motivation. Her clear rushing spirit often helped me to separate good from bad pearls and to dense and clarify theoretical ideas in this work. Prof. Ralf Der has initiated my interest in neural networks with his lectures on this issue followed by a programming project, with Ingmar Bauer on visual object identification for khepera robots. My colleagues from the Numerical-Toolbox Group at Bruker Daltonik Dr. Jens Decker, Dr. Michael Kuhn and Ute Clauss provided a helpful and motivating atmosphere and were open for questions and support in providing and preparing of proteomic data. Thanks also to Marc Gerhard, Marco Suesse, Karsten Bohne, Jens Rudolf and a lot of other guys from the ESW at Bruker for a cooperative team work. Prof. Herbert Thiele, Dr. Wolfgang Pusch, Dr. Markus Kostrzewa, Dr. Stefan Klebel and Dr. Thomas Elssner helped me to get a deeper insight in the expert knowledge for proteomic research and the specific aspects of data preparation of spectral data.

The workgroup of Prof. J. Thiery from the Institute of Laboratory medicine highlighted in a lot of fruitful discussions the clinical problems in analysing spectral data for a diagnostic purpose. Parts of this work are close to statistical research topics. In a collaboration with the Department of Statistics at the University of Dortmund the research group of Prof. Urfer has given helpful suggestions on statistical problems in the context of clinical proteomics. Hereby special thanks to Dr. Klaus Jung. Also the communications with Dr. Mathias Lindemann from the Center for Techo-Mathematics at the University of Bremen have inspired some contributions in this work. In addition, my thanks go to the *German Research Foundation* for providing foundings for conference participations.

Thanks for non scientific, social activities keeping up my motivation to Bettina, Matthias, Sophie, Nelly and Andreas from the *Tuesday Cinema Club* and Korinna and Tobias for humorous psychological meetings. My parents, of course were and are a main source of confidence and supported this work in all non scientific ways - thanks a million.

Leipzig, Oktober 2006 Frank-Michael Schleif

Contents

I Self-organization for spectral data **1**

1 Introduction **2**
1.1 Background and motivation . . . 3
1.2 Practical ways of clinical data analysis . . . 5
1.2.1 Curse of dimensionality . . . 5
1.2.2 Clinical data and model characteristics . . . 6
1.2.3 Long-time learning . . . 6
1.3 Mass spectrometry based clinical proteomics . . . 6
1.4 Scientific contribution and organization of this thesis . . . 10
1.4.1 Local relevance learning . . . 11
1.4.2 Fuzzy classification for Soft Nearest Prototype Classification . 11
1.4.3 Generalization bounds and parameter optimization . . . 11
1.4.4 Organization . . . 12
1.5 Notations, Abbreviations and Conventions . . . 12

2 Preprocessing of MS spectra **14**
2.1 Recalibration . . . 16
2.2 Baseline Correction . . . 17
2.3 Peak picking and feature extraction . . . 18

3 Unsupervised and supervised learning **21**
3.1 Unsupervised data analysis using PCA . . . 22
3.2 Vector Quantization . . . 23
3.3 Data analysis by SOM . . . 25
3.4 Neural GAS . . . 28
3.5 Estimation of the intrinsic dimension . . . 28
3.6 Supervised classification . . . 30
3.6.1 Bayes classification . . . 31
3.6.2 Linear and Quadratic discriminant . . . 31
3.6.3 Support Vector Machines . . . 33
3.7 Evaluation methods . . . 35
3.7.1 Cross-validation . . . 36
3.7.2 Sensitivity and specificity . . . 37

3.7.3 Receiver Operator Characteristic and Penalization 38

4 Supervised Neural Vector Quantization 41
4.1 Learning Vector Quantization 41
4.2 Generalized Learning Vector Quantization 44
4.2.1 Hypothesis margin 44
4.2.2 Penalization for GLVQ 45
4.2.3 Simulations 46
4.3 Supervised Neural GAS 47

II Self-adapted metric and label information 48

5 Relevance Learning in Learning Vector Quantization 49
5.1 Relevance Learning in GLVQ and SNG 50
5.2 SRNG and GRLVQ with Localized Metric Adaptation 53
5.3 Different examples of problem specific metric adaptation 55
5.3.1 Scaled Euclidean Metric in LVQ 55
5.3.2 Mahalanobis Metric in LVQ 57
5.3.3 Tanimoto Metric in LVQ 58
5.3.4 Correlative Feature Elimination with LVQ networks 59
5.4 Experiments and Results 61

6 Supervised Fuzzified Learning Vector Quantization 65
6.1 Self Adapted Soft Nearest Prototype Classification 65
6.2 Fuzzified Self Adapted Soft Nearest Prototype Classification 67
6.3 Relevance Learning in Fuzzified Self Adapted SNPC 68
6.4 Localized Metric Adaptation in FSNPC and SNPC 69
6.5 Interpolated Voronoi tessellation 69
6.6 Experiments 69

III Generalization theory and parameter optimization 71

7 Generalization Theory for Learning Vector Quantization 72
7.1 Introduction to Generalization Theory 72
7.1.1 Rademacher and Gaussian Complexities 73
7.2 Generalization ability of Generalized Learning Vector Quantization . 76
7.2.1 Generalization bounds for GRLVQ 76
7.2.2 Generalization bounds for GRLVQ with local Metric Adaptation 80

8 Heuristics for optimized Design and Learning 82
8.1 Active Learning Strategies 82
8.2 Dynamic Growing in Learning Vector Quantization 84
8.2.1 Insertion of prototypes 85
8.2.2 Deletion of prototypes 85
8.3 Experiments and Results 86

IV Soft Learning in MS Proteomic Research 95

9 Analysis of Mass Spectrometry Data from Clinical Research 96
9.1 Self Organizing Maps to visualize high-dimensional data 98
9.2 Unsupervised analysis on MS -data 99
9.3 Classification of MS -data 101
9.3.1 Classification with SRNG and Alternatives 102
9.3.2 Localized Classifiers for Multidimensional Distributions . . . 104
9.3.3 Supervised Learning with fuzzy Label Information 106

10 Conclusions 111
10.1 Summary 111
10.2 Open problems and future issues 112

A Derivations 114

References 131

Part I

Self-organization for spectral data

Chapter 1

Introduction

In the beginning the Universe was created. This has made a lot of people very angry and been widely regarded as a bad move.
Douglas Adams

High-throughput measurement technologies, such as microarray, cDNA and mass spectrometry, are changing the practice of biology and medicine. These methods are going to be used in different application fields such as the analysis of micro organisms [93], clinical diagnostics [37, 89], pharmaco-genomics [114, 30] and others. One main goal is the diagnosis of a disease and, more significantly, the classification of the type of disease. Moreover, therapies based on the disruption or mitigation of aberrant gene function contributing to the pathology of a disease should be developed [38].

These tasks involve the design of expression based classifiers with e.g. gen expressions in microarray or protein regulations in proteomics to discriminate differences in the cell state.

Classification methods of pattern recognition are clearly suited for diagnosis. They may also apply to therapy because prediction methods could be used to identify e.g. gene-gene and gene-phenotype relations in network modeling. Expression-based microarrays phenotype classification was one of the first of such applications and this led to a still growing amount of developments and publications [91, 89, 41, 180, 184, 110].

Microarray analysis yields a vector of gene expression levels which can be mapped to a class label as output by a trained classifier to predict the class of the input vector. Microarray analysis is by now a matured field of research. Proteomic profiling based on mass spectrometry on the other side is a relative new research field [183, 157, 180, 110] which does not focus on the gen level but on the peptide and protein level originating from the cell synthesis as a potential indicator for e.g. disease states. The analysis by mass spectrometry allows for a high throughput on reliably small costs. We are confronted with a vector of protein or peptide expressions[1] as input for a classifier and look again for a corresponding class label [78, 114]. Unlike gen expression data,

[1] high or down regulated

we thereby consider a spectral signal of proteomic data which potentially captures a larger amount of information originating from the cell.

Hence an effective and subtle diagnosis might become possible. Due to these complex data, preprocessing becomes a crucial point of classification methods and the identification of classification specific expressions is generally complicated. In the clinical context further requirements for a highly standardized quality control on each processing step [8] and for formally validated methods, have to be met. Thus, a classification which can be interpreted by categories as well as considerations about the confidence of the classification e.g. in terms of fuzzy values are required. This work extends prototype based self-organizing neural approaches in the processing of labeled, numerical vectorial spectral data under the special focus of clinical problems. Data processing refers to a large bunch of operations, like generation, storage, search and more advanced tasks such as extraction of information. In this work we will mainly focus on intelligent data analysis which on one side relies on an appropriate preprocessing of the data and on the other side on sophisticated methods for the generation of predictive models upon the prepared data. Thereby several improvements of self organizing neural networks which make the methods well suited for clinical proteomics have been proposed. The suitability of the paradigms is tested on real life data taken from clinical studies as well as on synthetic data.

1.1 Background and motivation

In clinical research the identification of biological features which allow to determine specific diseases is an important issue [38]. One way is offered by the characterization of the biological samples by use of mass spectrometry. This leads to very high-dimensional spectral data sets. The automatic analysis of such datasets is hard for ordinary data analysis algorithms as shown in [89]. On one side the preprocessing of the spectral data involves multiple complex steps as explained in the appendix, on the other side specific challenges occur for the subsequently applied statistical analysis. But also the estimation of a classifier model upon the preprocessed spectral data is still quite complex such that standard approaches may fail. Thereby especially the metric used in the high dimensional space to determine distances is a critical issue. Also the mathematical principles the algorithm is based on has to be robust to deal with high dimensional data. Approaches depending e.g. on density or probability estimations are sometimes even not applicable due to bad conditioned matrix operations for high dimensional data sets.

Further, in clinical studies a more detailed understanding of the data space becomes necessary and global approaches for data analysis are often not suitable [132]. For a classifier this means that a simple classification decision is not any longer sufficient, especially for a misclassification it should be possible to track the classification decision. Clinical data are typically complex allowing multiple labelings hence they are often multimodal and a classifier should be able to model this characteristic. In this

work especially self organizing neural network methods are considered, which have already proved to be beneficial for such kind of problems [166].

Many different types of artificial neural networks exist for data interpolation and classification. Early ideas were formulated by McCulloch and Pitts in 1943, who showed how simple models of nervous cell systems are able to perform logical computations [99]. In 1949, Hebb proposed a neural dynamic which integrated weighted input channels by means of summation nodes. Hebb used the additional assumption that if two of these simple computing units are simultaneously active, the weight between them is increased [68]. Suitably adapted variants of this Hebbian learning paradigm are found in many algorithms. Very prominent learning architectures are layered feed-forward neural networks (FNN) which propagate the input vector via weighted connections through internal layers of neural amplifiers to an output vector [67]. Learning takes place in a supervised manner by the adaptation of the connection weights according to the backpropagated error between the network output and the desired output [98]. This network type is known as universal approximator for any functional input-output relation, if enough units exist in the hidden layer(s) [69]. The training methods for the above approaches are variants of backpropagation, or general gradient descent techniques for minimizing the error. Kolen and Kremer give an overview of the methods in [85]. While FNN's are often successfully used in a lot of domains, they are less applicaple for clinical data problems. Here FNN's are often considered as *black box* systems which do not reveal much information about the decision process and hence are less interpretable, which is essential in clinical data analysis.

The path taken in this work is an alternative one: instead of training the connection weights between neurons of feed-forward networks, a more direct data representation is obtained by using prototype models. Prototypes are entities that can be adapted towards states encountered in the training set, which means that the similarity of a prototype is increased to what it stands for. Prototype representations store information in plain vector locations rather than in the forward-propagated, transformed network output states. Usually, the represented state is a prominent feature of the input data, and data points near a prototype are supposed to share the prototype's properties, if any given, e.g. a class membership. Different characteristic input states should be handled by different prototypes, therefore the number of prototypes is related to the granularity of the overall data representation. If no property or label is assigned, a single prototype should account for the fact that it is likely to find data in its neighborhood. The collectivity of all prototypes constitutes a rough sketch of the input space. After training, meaning can be manually associated with the found prototype locations.

The basic ingredients for a prototype model are (a) an appropriate metric in the input space in order to measure the similarity between data and prototypes, (b) a certain number of prototypes, and (c) a cooperation strategy to avoid the wasting of prototypes. The first point refers to the description of the data clustering, the second point to the model size, and the last point means that neighboring prototypes occupying the same location must negotiate a specialization precedence or a spreading scheme. With respect to the prototype-based data representation, the involved methods are predestined

to be using concepts of self-organization. On one hand, neural methods are called self-organizing, if they produce structured representations of the training stimuli. An explicit target function might not be given for the ordering process; still, the implicit goal might be a similarity-based mapping of the input to internal states, thereby reducing the adaptation efforts or the costs connected with learning. On the other hand, self-organization refers to competition or cooperation during the learning process: a neuron that is most responsive to a stimulus is selected as winner, and it takes much of the overall available update energy for specializing on the given input, whereas less matching prototypes are only slightly updated. Alternatively, several models can be trained on different aspects of the input data and linked together in order to synchronize unlabeled multi- sensory signals and to generate meaningful self-supervised states. Throughout this work, it is supposed that the natural redundancy of real life-data can be robustly captured by a relatively small number of prototypes.

The learning vector quantization (LVQ) for labeled data is determined by a supervised competitive winner-takes-all (WTA) dynamic by local updates implementing Hebbian learning. The main purpose is to solve classification tasks by exploiting certain data characteristics in order to emphasize similarities and differences: LVQ thereby learns distinct class labels. LVQ can be trained as an expert for an infinite set of decisions for given data. Thereby LVQ has been applied in very different domains and generalized in different ways [172, 58].

1.2 Practical ways of clinical data analysis

The methods derived in this work are in general applied to the analysis of clinical data. These data are mainly originating from proteomic mass spectrometry (MS) measurements and show specific properties and reveal special questions.

1.2.1 Curse of dimensionality

In most clinical studies the number of samples is relatively small[2]. This is due to different reasons such as the prior probability of the disease in the sample population, ethical or other non clinical restrictions. Each item is characterized by an in general large number of measurements. Where we have no or only small additional knowledge about the statistics of the data. The specific distributions of the data are not known. Further we are confronted with the curse of dimensionality ([66] p. 22f). Typically, the measured spectral data consist of around 10 to 45 thousand measurement points per sample. Thereby the clinical relevant information is encoded only in a small number of measurement points (typically around $1 - 100$ dimensions) which are in parts related to each other by dependences orginating from the functional character of the spectra as well as due to biochemical reasons.

[2] In general an initial study starts with only 10-20 samples per class.

1.2.2 Clinical data and model characteristics

For clinical data the labeling of the data or the specific class/label is subject of uncertainty especially if different disease stages come into play. Therefore a model should be able to predict the class responsibility with a specific degree of safety insuch a way that a new item belongs very likely to the first two stages/classes but not to the remaining classes.

Further, the determined models have to be of very high precision (less than $1-5\%$ misclassification) and should be as clear as possible (easy to understand)[3]. For reduction of measurement variance it is possible to measure replicates of the same sample, which is typical in mass spectrometry based clinical proteomics. This gives an artificial increase of the number of items and has to be considered in the model estimation process. Especially the model generation should not become inaccurate or slow down due to replicates. There are different ways to deal with replicates. Replicates can be averaged to reduce the measurement variance, a single or some realizations of a measurement can be selected using some quality criterion or all replicates can be used. In the later case the crossvalidation should take care on that fact and the classifier should not slow down during training due to multiple, similar measurements of the same sample.

1.2.3 Long-time learning

Often, the initial number of samples in a clinical analysis maybe small. However it can happen that the sample set grows during time due to continued clinical measurements. As long as new samples are prepared in the same manner as before it would be beneficial to incorporate the new knowledge into existing models and hence an adaptive learning scheme is recommended. This is known as long-time learning and a typical property of prototype based classification methods. Most standard approaches determine models which are not adaptable and need a complete regeneration if the sample set is changed. Under this constraints it is obvious that most standard approaches do not fit and alternatives have to be considered.

1.3 Mass spectrometry based clinical proteomics

Mass spectrometry is an analytical technique used to measure the mass-to-charge (m/z) ratio of ions. It is most generally used to find the composition of a physical sample by generating a mass spectrum representing the masses of sample components. This technique can be used for different kinds of analysis. In this contribution we will focus on clinical proteomics. Thereby a mass spectrometer is a device that measures the mass-to-charge ratio of ions. This is achieved by ionizing the sample and separating

[3] Both are typical requirements for all clinical studies.

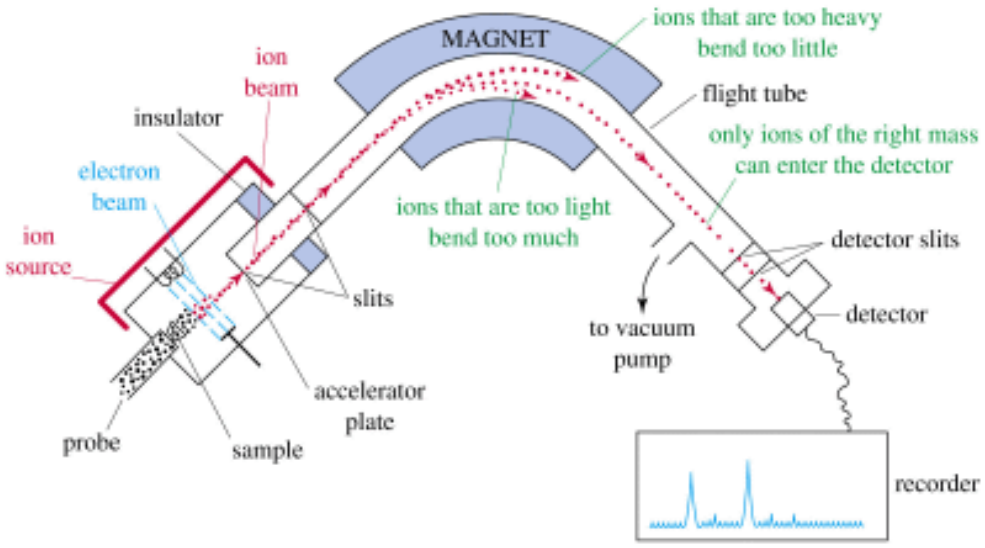


Figure 1.1: Principle function of a mass spectrometer taken from [178].

ions of differing masses and recording their relative abundance by measuring intensities of ion flux. A typical mass spectrometer comprises three parts: an ion source, a mass analyzer, and a detector system [178]. The principle is depicted in Figure 1.1 Different technical realizations of mass spectrometers are possible. A good overview and explaination is given in [178].

The principle is based on the fact, that different chemicals have different atomic masses. This is used in a mass spectrometer to determine what chemicals are present in a sample. For example, NaCl is turned into gas and ionized (broken down) into electrically charged particles, called ions, in the first phase of the mass spectrometry. The sodium ions and chloride ions have specific atomic weights. They also have a charge, which means that their path can be controlled with an electric or magnetic field. The ions are sent into an acceleration chamber and passed through a slit in a metal sheet. A magnetic field is applied to the chamber. This field, according to Figure 1.1, can be imagined as the north being located under the mass spectrometer, while the south end of the magnet being located in front of the picture. The field pushes each ion perpendicular to the plane defined by the particles direction of travel and the magnetic field lines. They are then deflected (makes them curve instead of traveling straight) onto a detector. The lighter ions are deflected more than the heavier ions. The magnetic field can push the lighter ions further, thereby giving them a larger deflection, than the heavier ions. The detector measures exactly how far each ion has been deflected, and from this measurement, the ion's 'mass-to-charge ratio' can be worked out. From this information it is possible to determine with a high level of certainty the chemical composition of the original sample [178].

For the analysis of proteomic data two specific realizations of mass spectrometers are relevant. One is *Surface Enhanced Laser Desorption Ionization* (SELDI) mass

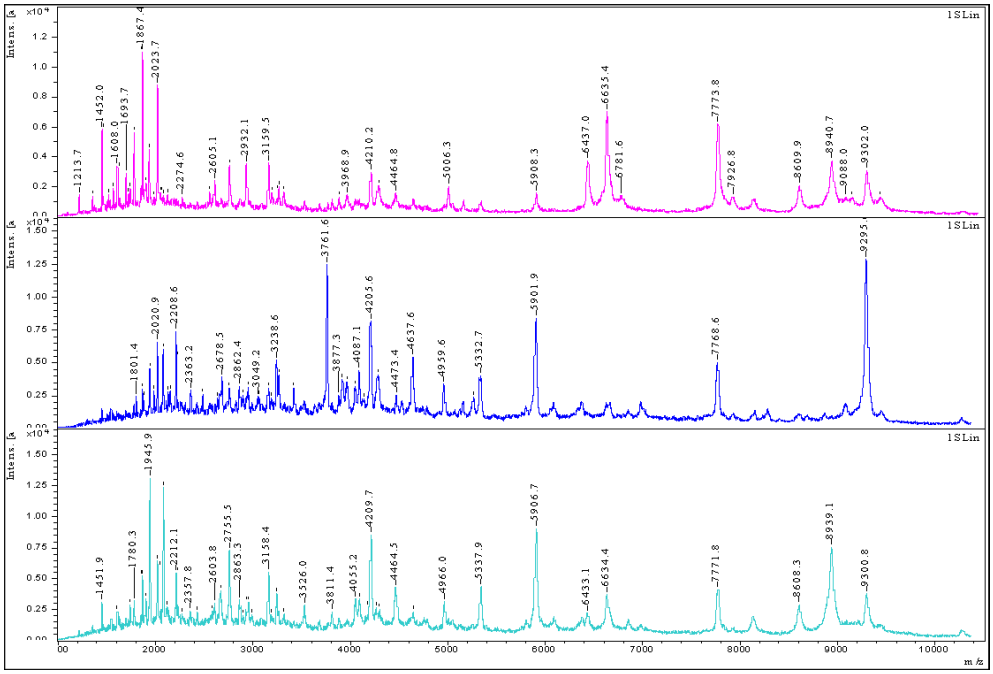


Figure 1.2: Spectra obtained from MALDI-TOF measurements. The spectra are from the same sample by use of different chemical preparations. Its obvious that the spectra contain specific attributes and hence the finally obtained sets of features cover different characteristics of the biological sample. The different preparations allow multiple screenings in the search for biomarker candidates. The data are measured in a common mass range of 1-10kDa with approximately 45000 measurement points. The marked position (mass labels in the plot) indicates specific peaks which have been used for further analysis. These still high dimensional data ($\approx$ 100 dimensions) are obtained after a series of preprocessing steps as explained in Chapter 2, giving so called peak lists.

spectrometry (MS) and the other one is Matrix-assisted laser desorption/ionization (MALDI) MS. In this contribution we consider data which are mainly obtained by MALDI-MS measurements. But the methods can be used for SELDI data too. For details on both methods we refer to [178] and [90]. Details on the preprocessing of the data used in this thesis can be found in Chapter 2.

The obtained spectral data for proteomic measurements of MALDI-MS data are similar to the data depicted in Figure 1.2 There also the peak lists are shown which are used in the following data analysis tasks. Subsequently some generic definitions

Definition 1 (Proteomics) Proteomics *is the study of the proteom, the protein complement of the genome. The terms* proteomics *and* proteom *were coined by Marc Wilkins and colleagues in the early 1990s and mirror the terms* genomics *and* genome, *which describe the entire collection of genes in an organism [90].*

Data set	Long name	Parameters
WDBC	Wisconsin Breast cancer data	two classes, 569 samples, 30 dimensions (dim)
NCI	NCI Prostate cancer data	SELDI-TOF, four classes, 222 samples, 130 dim.
EVMS	East Virginia Medical School	SELDI-TOF, four classes, 196 samples, 82 dim.
LEUK	Leukaemia	MALDI-TOF, two classes, 154 saples, 145 dim.
Proteom 1	Bruker 1 (colorectal cancer)	MALDI-TOF, 3 classes, 203 samples, 78 dim.
Proteom 2	Bruker 2	MALDI-TOF, 2 classes, 94 samples, 124 dim.

Table 1.1: Cumulative listing of the different MS data sets and their measurement parameters

Definition 2 (Matrix-assisted laser desorption/ionization) Matrix-assisted laser desorption/ionization *(MALDI) is a soft ionization technique used in mass spectrometry, allowing, among other things, the ionization of biomolecules (biopolymers such as proteins, peptides and sugars), which tend to be more fragile and quickly lose structure when ionized by more conventional ionization methods. The ionization is triggered by a laser beam (normally a nitrogen-laser). A matrix is used to protect the biomolecule from being destroyed by a direct laser beam [178].*

Definition 3 (Time-of-flight) *The* Time-of-flight *(TOF) method of measuring particle mass-to-charge ratio is done as follows. An ion of known electrical charge and unknown mass enters a mass spectrometer and is accelerated by an electrical field of known strength. This acceleration results in any given ion having the same kinetic energy as any other ion given that they all have the same charge. The velocity of the ion will depend however on the mass-to-charge ratio. The time that it subsequently takes for the particle to reach a detector at a known distance is measured. This time will depend on the mass-to-charge ratio of the particle (heavier particles reach lower speeds). From this time and the known experimental parameters one can find the mass-to-charge ratio of the particle. This method of analysis is a powerful tool for finding the mass-to-charge ratio of charged particles, atoms and molecules [178].*

For the most common kinds of analyses the given biological samples and measured data are too complicated to be processed from scratch, some preprocessing has to be done. In this work parts of the data are synthetically generated in accordance to the considered problems and parts of the experiments were done using data from public databases such as UCI [13] and NCI [42]. The remaining data originate from own mass spectrometry measurements on clinical proteomic data[4]. Measurement details for the LEUK data are given in [101]. The individual data sets are listed in Table (1.1) with their corresponding parameters[5].

The MALDI-TOF data originate from measurements of particular treated biological samples. The biological samples (e.g. blood of cancer or control patients) are pretreated by a bio-chemical eluation procedure which allows the specific capturing of

[4]The data measured on Bruker systems are confidential so only general characteristics are listed here.

[5]Only the main parameters are shown.

parts of the sample material (e.g. extraction of cupper affine chemical compounds). Some details on the used procedure are given in Chapter 2. The obtained eluates are, after some further steps, measured by a MALDI-TOF MS. The obtained raw data of this MS measurements are spectral data with measurement points at m/z and corresponding intensity values I. The spectral data consists in general of 45000 measurement/sample points (sp) measured in a range of $1kDa$ to $10kDa$. For a detailed explanation of the technical measurement procedure a good introduction is given in [90]. The determination of a classification model to differentiate samples from one class with respect to another class requires some additional mathematical processing steps on the MS spectra. These steps include the substraction of underground artefacts referred as baseline, the recalibration or alignment of the spectra to each other to make them comparable and the application of a peak picking procedure to get the numerical features used in further processing. The features considered in this work can still be tracked back to the original data and allow a local analysis of the data which is important for biomarker research. The details on the preprocessing steps are given in Chapter 2. Important is, that the obtained features are *peak areas*, which are normalized in $N(0, 1)$. Such, the original data space is reduced to at least 100 dimensions without a significant loss of information, keeping the relation to the functional mass spectrometry data behind.

Beside of mass spectrometry based clinical proteomics also other measurement techniques maybe used to analyzed proteomic samples. A good overview can be found in [90].

1.4 Scientific contribution and organization of this thesis

The structural simplicity of the basic LVQ algorithm and its intuitive understanding make it a good starting point for self-organizing vector quantization in the domain of clinical questions. LVQ algorithms fit already some of the needs, which occur in clinical proteomics such as simplicity and good interpretability. However some other attributes required for clinical studies can not be handled by LVQ networks so far. In clinical solutions a high interpretability of the classifier models is needed to track the decision process. Clinical spectral data are in general multi modal and are better represented using multi modal approaches, thereby the specific discrimination properties of potentially hidden sub groups of the data with respect to the remaining data are of special interest. To deal with the formerly mentioned challenges new LVQ based algorithms have been developed. Further clinical data are subject of fuzziness with respect to the given class label and a classification decision upon the given set of decision parameters. This should be taken into account by use of a machine learning approach. In addition clinical studies typically start with a relative small set of individual measurements (pilot study) which will increase in a larger clinical project or by

additional measurements over time. Hence it would be desirable if the classification model is adaptable to new data avoiding a full remodelling. The new approaches have been formely derived and applied on synthetical as well as on clinical data sets.

1.4.1 Local relevance learning

In the first part of this thesis specific kinds of LVQ networks have been extended by an alternative cost function to support local relevance learning. Different kinds of metric adaptations together with experimental results are reported. The original formulation of relevance adaptation for LVQ in terms of the *Generalized Relevance Learning Vector Quantization* (GRLVQ) date back to earlier work of HAMMER AND VILLMANN [62]; however the principle has been theoretically and practically improved to local concepts and alternative metrics by the author and have been published in a number of publications: the papers can be split into extensions of GRLVQ and *Soft Nearest Prototype Classification* (SNPC) to local metric adaptation [132, 131, 130], general metrics [172] and applications [171, 165, 125, 19, 129]. The SNPC model has been improved by a new cost function incorporating local relevance learning and has been published from theoretical [131, 173, 130] as well as from practical points of view [132, 130]. The local relevance learning as well as the different kinds of alternative metric adaptation allow a better modeling of the classification problem and an alternative view on the data revealing new insights in the unkown data specifics.

1.4.2 Fuzzy classification for Soft Nearest Prototype Classification

In clinical proteomics the underlying data distributions as well as the corresponding labeling of the data is subject of fuzziness. Hence it is desirable to use algorithms which are able to support unsafe classification decisions. The SNPC model has been improved by a new cost function incorporating fuzzy classification and has been published in [131, 173, 130]. The approach is suited to indicate the fuzziness of the decision process which is especially interesting for overlapping data distributions.

1.4.3 Generalization bounds and parameter optimization

In the second part of the thesis, theoretical aspects on generalization bounds for local prototype based classifiers are derived [61, 56] and new growing and active learning concepts [127, 128] are developed based on the theoretical results. This allows an effective parametrization of the algorithms making them easier applicable in the clinical context. By the active learning strategies the long-time learning paradigm becomes available.

1.4.4 Organization

At the beginning a review of the different preprocessing steps applied on the spectral data sets is given. Subsequently an introduction into the different known machine learning concepts used in the later derived methods is given. Followed by a short review of methods used to evaluate the obtained results. Different improvements for relevance learning are considered and local metric adaptation is introduced. Then a specific prototype based classifier is improved by fuzzy classification and local relevance learning. The third part derives theoretical generalization bounds for the localized relevance paradigm closed by some techniques for parameter optimization. In the last part the different methods are applied on real life data in an extensive analysis showing the usefulness of the introduced extensions.

Before specific learning architectures are discussed, some notation conventions will be agreed on.

1.5 Notations, Abbreviations and Conventions

A *pattern* is equivalent to a *data point*, which may be represented by a *weight vector* of a *prototype* neuron. In the context of proteomic research and biomarker identification a pattern may further refer to a subset of the input variables which had been identified to be characteristic for the underlying classification. The prototype closest to the currently presented pattern is referred to as the *winner*, regardless of its represented class. As usual, a *cycle* is an epoch referring to one presentation of all data patterns in a training set; this must not be confused with *iteration* which describes the presentation of only a single pattern from the training set. The most common symbols in this work are listed in Table (1.2).

Symbol	Meaning
V	data space
D_V	data space dimension
$\mathbf{v}, \mathbf{x}$	data point
$\mathbf{w}$	prototype
W	set of prototypes
W_c	set of prototypes with class label c
c_v	label of input $\mathbf{v}$
V_c	subset of V with data of class c
$\mathcal{L}$	set of labels
$N_{\mathcal{L}}$	number of labels
N_S	number of samples
N_c	number of samples with class label c
N_W	number of prototypes
K_c	cardinality of W_c
Ψ	mapping function (e.g. winner takes all)
$\{\ldots\}_+$	entity with the current (correct) label
$\{\ldots\}_-$	entity with an alternative (incorrect) label
$d(\mathbf{v}, \mathbf{w})$	distance function (Euclidean)
$d_{\mathbf{r}}(\mathbf{v}, \mathbf{w})$	parametrized distance (weighted Euclidean)
$t(\mathbf{v}, \mathbf{w})$	distance function (Tanimoto)
$m(\mathbf{v}, \mathbf{w})$	distance function (Mahalanobis)
E	energy/cost function
L	loss function
h	neighborhood function
g	prediction function (classifier)
$p(\mathbf{v})$	density distribution
λ	relevance profile vector
λ_i	dimension relevance with index i or eigenvalue to eigenvector i
$\mathbf{\Lambda}$	matrix of relevance vector (e.g. for local distances)
ϵ	learning rate
MS	mass spectrometry
m/z	measurement per time
I	if not stated otherwise - intensity
Da	m/z is measured in Dalton
TOF	time of flight
SELDI	surface enhanced laser desorption ionization
MALDI	Matrix assisted laser disruption ionisation
LVQ	Learning Vector Quantization
GLVQ	Generalized Learning Vector Quantization
GRLVQ	Generalized Relevance Learning Vector Quantization
SRNG	Supervised Relevance Neural GAS
SNPC	Soft Nearest Prototype Classification
LSNPC	Localized SNPC
FSNPC	Fuzzy SNPC

Table 1.2: Abbreviations and symbols

Chapter 2

Preprocessing of MS spectra

In most cases a given data set is too raw to be processed from scratch, some preprocessing has to be done. The obtained raw data of these MS measurements are spectral data with measurement points at m/z and corresponding intensity values I. For a detailed explanation of the technical measurement procedure a good introduction is given in [90] the generic procedure is shown in Figure (2.1). Subsequently the adapted

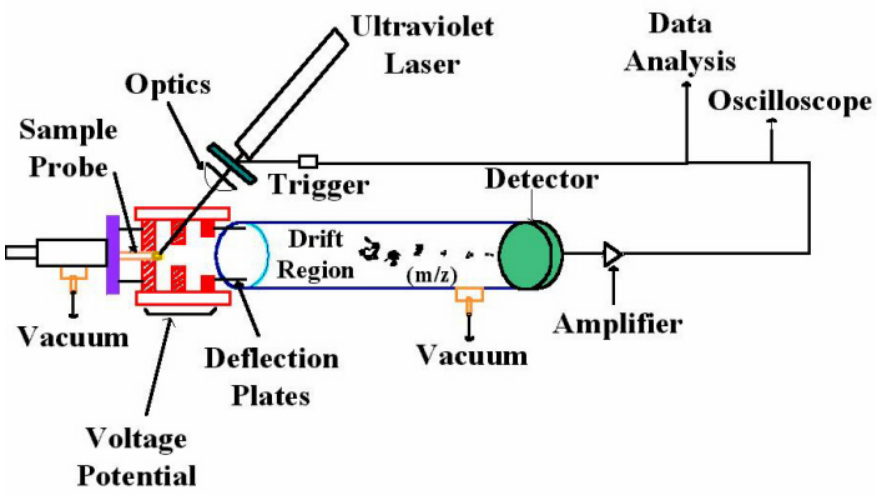


Figure 2.1: Schematic figure of the main parts of a MALDI-TOF device [97]

procedure as it has been used in this work is explained. At the beginning we start with a sample of e.g. blood-serum, plasma, urine or some other body fluid. This sample consists in general of a complex compound of chemical and biological substances, which are mainly irrelevant for the analysis in focus. Therefore the sample undergoes a chemical preparation step at the beginning to extract specific compounds, which are considered to be problem specific by some preanalytic knowledge. For MALDI-TOF this can e.g. be done by use of the so called magnetic bead technology [157]. We will not go into details here but give a short glimpse on the principle behind (c.f. Figure 2.2). The magnetic beads are magnetic particles with specific surface properties such

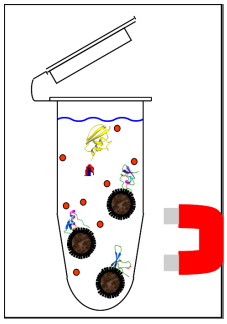


Figure 2.2: Principle scheme for bead base preparation of biological samples.

that specific substances can bind on receptors upon the bead surface. For example, we may consider beads, which are affine to cation's (e.g. WCX - cation exchanger) and allow corresponding bindings in the sample fluid. These bindings occur in general at a certain Ph-level and can be resolved at another one. By use of these properties the beads are given to the sample at the binding Ph-level and intermixed, followed by a centrifugation. Because of the magnetic properties of the beads the experimenter is now able to separate the beads with the binded structures from the remaining sample material. In a subsequently washing procedure the beads and binded structures can be eluated. Then we end up with the separated biological compounds, which are believed to be relevant for the subsequent analysis.

Now this material is applicated to a specific matrix material. The matrix protects the biological material from destruction during the laser measurement and consists of different polymer compounds. The purpose of the matrix is to absorb the main energy of the laser such that only a specific amount of induced energy is transmitted to the sample compounds embedded in the matrix. Due to the induced energy the compounds are emitted from the matrix and fly in the drift region to the detector on the opposite site.

Thereby the compound is fractioned into its elementary atomic structures. Considering the laser shot direction, the induced energy and the flight time of the fractionated elements measured by the detector, one is able to determine the specific mass of each element. Hence, we end up with a measured spectrum of intensity values as a relative amount of the element in the sample at a specific mass/time (mz) position. This is shown in Figure 1.2. Thereby we find that the spectra differ with respect to the used sample preparation method (here different bead particle types) as depicted in Figure 1.2.

The spectral data consist in general of 45000 measurement/sample points (sp) measured in a range of $1kDa$ to $10kDa$. Each spectrum is coupled with an underlying structure due to the evaporated matrix and other material called baseline. In addition spectral data, which belong to a common class, have to be aligned to each other, due to small shifts in the measurement. These two steps are known as *baseline correction* and *recalibration* and are explained very briefly in the following.

If these elementary steps are done one is still confronted with an e.g. 45000 dimen-

sional feature space. The general expectation, however, is that the number of relevant input dimensions is around $1 - 100$ dimensions. This is reflected by biological and bio-chemical expert knowledge, because only a limited number of data in a spectrum is important. Hence, it is necessary to apply a *peak detection* algorithm upon the given spectra, which is also explained in the following in short. From the picked peaks so called *peak areas* are calculated and normalized in $N(0, 1)$ which are finally used as features. Such, the original data space is reduced to less or some 100 dimensions without a significant loss of information. A detailed explanation of all processing steps can be found in [14].

2.1 Recalibration

If a set of spectra is measured one can find small shifts also in measurements of the same sample. These shifts are measurement dependent and can be explained by a linear or quadratic error model. The recalibration problem is shown in Figure 2.3 were a number of spectra before and after recalibration is shown. The recalibration or sometimes alignment is necessary to get a comparable set of spectra were common masses in different spectra are mappable to each other.

Each spectrum consists of a sequence of intensity values $(y_i)_{i=1}^K$. These values are recorded at discrete flight time values $(t_i)_{i=1}^K$, where typically $t_i = t_0 + i \cdot t_b$ with constants t_0, t_b. A second mapping is required to assign m/z values to the data points. In our case we use the following calibration function (2.1):

$$t = c_0 + c_1 \cdot \sqrt{(m/z)} + c_2 \cdot m/z \tag{2.1}$$

with constants c_0, c_1, c_2. These constants are determined by some calibration measurement with known calibration masses $(M_j)_{j=1}^{M_C}$ (we assume z=1 for simplicity). Such a measurement yields pairs $(t_j, M_j)_{j=1}^{M_C}$, where t_j is the observed time of flight of mass M_j. The constants c_0, c_1, c_2 are the solution of a least square fit for Equation (2.1).

In practice, systematic mass shifts can be observed for individual measurements. An upper estimate e_{shift} of such shifts can be given, e.g. $e_{\text{shift}} = 1000$ ppm. Such shifts are caused, e.g., by the varying height profile of the spots on the targets and cannot be computed in advance. That is why a recalibration (alignment) of the spectra is required. For that purpose individual values of c_0^i, c_1^i are computed for each spectrum, a re-fit of c_2 is not necessary. The recalibration procedure consists of two steps:

1. determine a list of reference masses $(M_j)_{j=1}^{M_R}$,
2. recalibrate the individual spectra using this list.

Both steps are based on peak lists $(m_k^i)_{k=1}^{M_R^i}$ and the corresponding flight times $(t_k^i)_{k=1}^{M_R^i}$ for each spectrum (i), $i = 1, \ldots, N$. An appropriate pre-selection of peaks, e.g. using an intensity threshold, is being applied. The list of reference masses is obtained as follows:

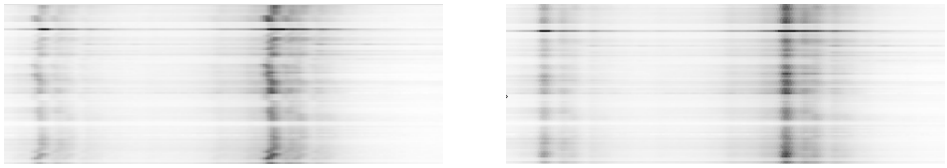


Figure 2.3: Not recalibrated spectra in a top view left. Spectra after recalibration right. Dark regions indicate peaks.

1. Using all spectra, find subsets of peaks such that

$$\max \text{Mass} - \min \text{Mass} < e_{\text{shift}}$$

   within all subsets.

2. Selection of all subsets with more than $N/3$ elements.

3. M_j constitutes the average mass of subset j.

Finally this list is used to recalibrate the spectra:

1. Select peaks (m_k^i) of the spectrum (i) such that $|m_k^i - M_j| < e_{shift}$ for some j,

2. Taking pairs (t_k^i, M_j) to compute c_0^i, c_1^i by a least square fit.

In practical applications one usually find about 40 reference masses, and about 15-20 peaks are used for the recalibration of the spectra.

2.2 Baseline Correction

During the measurement of a spectrum the real spectrum is overlayed due to electronic noise or artefacts from the matrix preparation by some underground named as baseline (c.f. Figure 2.4). This underground should be removed to avoid errors in subsequently feature extraction steps. Different algorithms for baseline correction of spectra, or MS -spectra in particular, have been proposed so far [124, 174, 111]. The algorithm used for the baseline correction here, is based on the *top-hat* filter, which subtracts the morphology opening operator [137, 124]. An efficient implementation of the required erosion and dilation operators can be found in [142]. The erosion operator for a flat structuring element (SE) replaces each intensity value by the minimum of the intensity within $\pm n$ data points around the very data point. The following dilation operator is replacing the values by the maximum within $\pm n$ data points. The width of the SE used in the opening operator is usual constant for the whole spectrum. The used implementation[1] is based on the contribution in [142] but with variable width of the structuring element to incorporate the change of the peak width.

[1] part of the ClinProt-System, Bruker Daltonik GmbH, Bremen, Germany

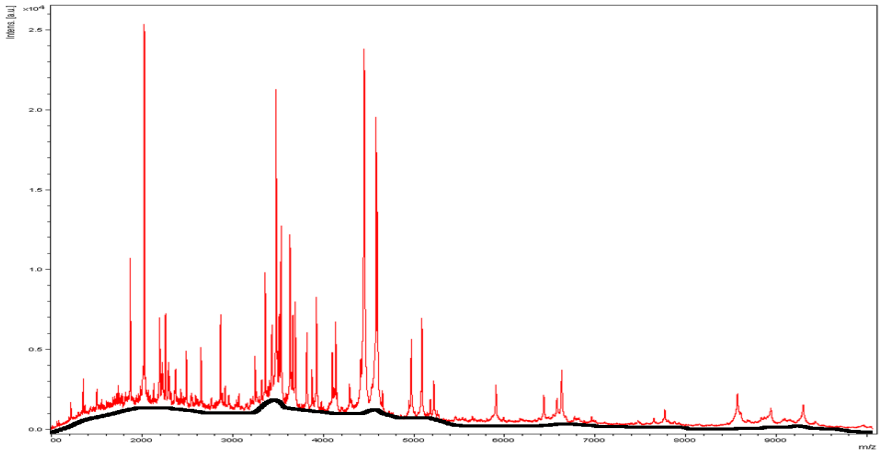


Figure 2.4: Typical spectrum with an underlying baseline - indicated by the dark diagonal line

The width w_m of the SE in m/z units for linear TOF spectra is calculated from the m/z value m according to the empirical formula

$$w_m = \max\{c_1 \cdot m, \min\{\frac{m}{4}, eq_{\text{fit}}\}\}$$

with

$$eq_{\text{fit}} = \max\{c_2, c_3 \cdot \ln(m) - c_4 + c_5 \cdot m - c_6 \cdot m^2\}$$

and constants

$$c_1 = 0.1 \quad c_2 = 40.0 \quad c_3 = 91.15$$
$$c_4 = 445.68 \quad c_5 = 2.55 \cdot 10^{-2} \quad c_6 = 1.01 \cdot 10^{-7}$$

The width w_m is then transformed to an odd value width in data points by means of the calibration function of the spectrum. The constants $c_3 \ldots c_6$ have been obtained from averagine peptides. They have been linearly scaled to give a width of the SE, which is large enough to bar the baseline from growing into the peaks of the spectrum. c_1 and c_2 define a minimum width of the SE.

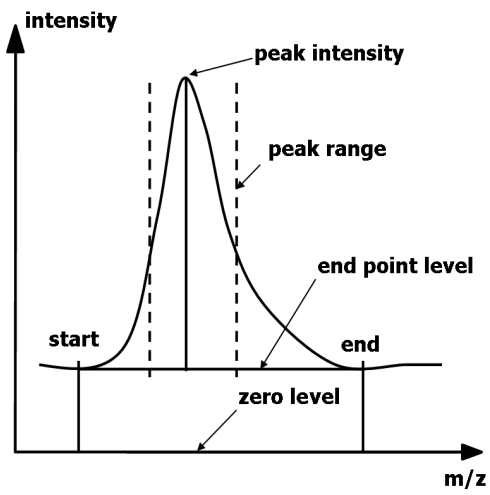
2.3 Peak picking and feature extraction

To obtain the final features and to significantly reduce the feature space the recalibrated and baseline corrected spectra are further processed using a peak picking procedure. Different approaches for peak picking can be found [103, 150, 111, 182]. Thereby a peak in a spectrum indicates a local maximum in the signal with a specific width related to the local position. In addition criteria such as a signal to noise ration (S/N) are used to separate small artefacts from a real peak [111]. From a biochemical point of view a peak in the mass spectrum indicates an underlying biochemical compound at a specific mass position. Typical characteristics of a peak are depicted in Figure 2.5(a).

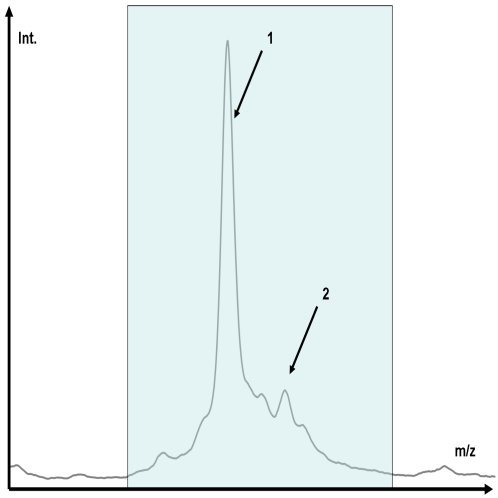
Thereby an average spectrum of the aligned spectra is calculated (a similar approach is explained in [103]). Upon this spectra a local peak detection algorithm is applied to identify regions, which can be considered as peaks. Peak picking in general is a complex processing task. Aspects such as the resolution of the device and an appropriate noise estimation must be dealed with. To get reliable results a number of assumptions is undertaken. On one side it is assumed that the average spectrum is a suitable approach to identify sufficiently stable peaks over the set of spectra, on the other side some parameters are estimated to setup the peak picking algorithm. The details are very specific and are not explained here in detail, only a raw sketch of the approach is given. The measurement procedure gives a first hint on peak characteristics. In linear mode the resolution of the mass spectrometry measurements is limited. In general for MALDI-TOF's one get resolutions within 500-800 Da. Thereby resolution stands for a measurement precision of the device with respect to the measured mass.

Resolution Several different definitions of resolution are used in mass spectrometry. In this work we focus on MALDI-TOF MS and hence we consider resolution as in the 50% peak-height definition.

On a low resolution underlying masses are not sufficiently resolved and different masses may be covered by a single peak. A high resolution in contrast allows, that also close masses can be identified in the spectrum by individual peaks (c.f. 2.5(b)). Knowing the resolution in advance tells us something about the peak width, which can be expected. Further for higher masses the width of the peak is increasing, which may also be incorporated into the picking procedure. To realize the peak picking multiple first deviation of parts of the average spectrum are calculated and a minimum search is applied. This procedure is done in an iterative local procedure considering only parts of the spectrum. Thereby knowledge about the expected peak width and a local noise estimation is incorporated. From the estimated peak regions a number of parameters such as start, end position, maximal intensity and 50% peak as well as a local resolution are calculated (c.f. 2.5(a)). The determined ranges are applied on all individual spectra to get the individual features as peak areas integrated locally within the given start and end positions for each spectrum [14].



(a) Typical characteristics of a peak in a mass spectrum. The figure shows a schematic drawing of a typical peak curve with mass position on the x-axis in (m/z) and corresponding intensity values at the y-axis. Different potential features can be extracted from the peak curve. Typically the peak intensity or peak area are used. These values maybe calculated in two kinds by zero level calculation or end-point level calculation. Zero point level means that a peak which has an offset in the intensity (as shown in the graph) is considered without this offset. End point levels means that the offset is ignored and start and end-point of the peak are connected by a line. This line is considered as the ground of the peak. In our analysis the zero point level was used for calculations



(b) The figure shows a broad peak, which in fact consists of probably $2-4$ peaks. The peak indicated by '1' is clearly resolved and shows a good spike. The region indicated by '2' may hide further peaks, which could not be sufficiently resolved. Parts of the energy of the unresolved peaks are accumulated by peak '1'. Unresolved peaks are a typical source of variance in the experimental design. In parts this problem can be solved by smoothing strategies or by use of more complex peak models such as sparse approximations or approaches, which model a peak by superposition of different generic functions (e.g. Gaussians).

Figure 2.5: Schematic peak characteristics and a common peak resolution scenario

Chapter 3

Unsupervised and supervised learning

Data analysis and data preprocessing are dependent on each other and a sufficient data analysis typically needs appropriate preprocessing steps. Especially for high dimensional data it is often necessary to apply data processing steps incorporating a high amount of data dependent expert knowledge. For mass spectrometry data the preprocessing steps are specific with respect to the measurement device and the physical measurement procedure. A large number of preprocessing steps can be applied. For the considered data in this thesis the necessary procedure is explained in Chapter 2.

The primary task of preprocessing is to extract that part of the data, which is important for a subsequent analysis step. In addition, preprocessing can be used to smooth the data and to reduce the number of dimensions to make a further processing more tracktable. Thereby preprocessing is typically unsupervised which means that no additional meta information as e.g. the class labeling is incorporated into the preprocessing.

Besides data preprocessing also the initial analysis of the data may focus on an unsupervised approach. Thereby the data are mainly described by their statistical properties which are believed to discover data specific knowledge. Different approaches for unsupervised data analysis may be considered. Common methods are Principal component analysis (PCA) [74], Self Organizing Maps (SOM) [83], Sammon Mapping [120], Multi dimensional scaling (MDS) [152, 80] or various clustering approaches [72].

Thereby clustering can be defined as:

Definition 4 (Clustering) *Clustering is the partitioning of a data set into subsets (clusters), such that the data in each subset (ideally) share some common trait - often proximity according to some defined distance measure.*

We will not go into details for these methods but focus on PCA, SOM and a specific kind of clustering approach, important in the subsequent data analysis in this thesis.

3.1 Unsupervised data analysis using PCA

A typical approach in the unsupervised analyis of high dimensional data is given by the *Prinical Component Analysis* (PCA). A detailed explanation of PCA can be found in [74, 66]. Here we only give a brief description. The PCA aims on detecting regularities in an unlabelled set $S = \{\mathbf{x_1}, \ldots, \mathbf{x_{S_N}}\}$ of points from $V \subseteq \mathbb{R}^{\mathbb{D}_\mathbb{V}}$. This task is realized by finding a low-dimensional representation of the data such that the expected residual is as small as possible. The aim is to find a smaller set of variables defined by functions of the original features in such a way that the data can be approximately reconstructed from the new coordiantes. For simplicity we assume a linear relation between the variables, non linear extensions are available e.g. by non-linear PCA as presented in [106]. For linear functions the problem is equivalent to projecting the data onto a lower-dimensional linear subspace in such a way that the distance between a vector and its projection is small [140]. The problem of minimising the average squared distance between vectors and their projections is equivalent to projecting the data onto the space spanned by the first k eigenvectors (v) of the matrix $X'X$ (sample covariance matrix)

$$X'Xv_i = \lambda_i v_i \qquad \lambda_i \text{ eigenvalue to } v_i$$

and hence the coordinates of a new vector $\mathbf{x}$ in the new space can be obtained by considering its projection onto the eigenvectors $< \mathbf{x}, \mathbf{v_i} >, i = 1, \ldots, k$. In other words the objective of the PCA is to determine so called principal components (PC) which are in fact the Eigenvectors orginating from a Single Value Decomposition (SVD) of the covariance matrix of the original data space. Thereby the first PC direction has the property to explain the highest variance among all normalized linear combination of the input dimensions in the original data dimensions and the last principal component the lowest variance respectively.

The PCs can be further analysed with respect to so called loadings of the original data dimensions contributing to a single PC [66]. These loadings can be interpreted as a ranking of relevance of the corresponding variables within the construction of a single PC. By use of this ranking the PCA can be used for dimension reduction of feature selection. Typically dimensions with a low contribution to the first principal component are abondend such that the data dimension can be significantly reduced as long as the variance in the data is sufficiently explained by a small amount of PC in a small set of variables from the original data space (c.f. 3.1). Thereby the data reduction is done only in an unsupervised scheme ignoring any additional information as e.g. label information given to the data. Nevertheless for a larger set of supervised analysis PCA has been incorporated in combination with e.g. a linear classifier to build a classifier on a reduced data space.

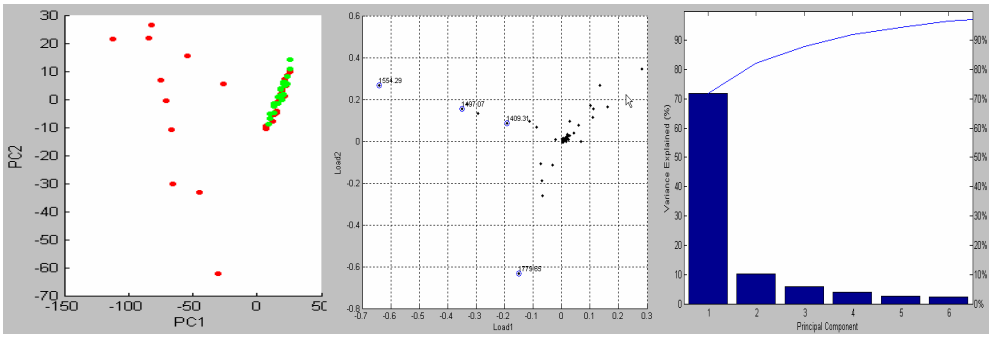


Figure 3.1: The figure shows typical results from a PCA analysis on a high dimensional MS-spectral data set (PC1 vs PC2 plot, Loading plot, Explained variance plot). The data are given in two classes (control, cancer) with 50 samples per class. For proteomic data a PCA analysis gives in general no sufficient separation between the different classes. This is due to the generic effect that the projection with respect to the explained variance is not characteristic for the underlying class separation problem. In the PC1 vs PC2 plot this can be seen because the points corresponding to the control class have a $PC1 < 0$ and are covered by items of the cancer class, which has another second cluster for $PC1 >> 0$. The loading plot of the PCA gives an indication which variables from the original data space (peaks) explained the most variance. In general a large number of variables have loadings of around 0.0 and only a smaller number did gives larger contribution to the PC calculations. Also for this analysis the identified peaks are often not important for a supervised analysis. The plot of explained variance indicates that PC1 already explaines 70% of variance and with the PC1-PC3 already $\approx 90\%$ of variance are explained. Hence the variance in the data is not spread about a larger number of principal components.

3.2 Vector Quantization

In both neural maps as well as vector quantization, these networks project data from a possibly high-dimensional input space $V \subseteq \mathbb{R}^{D_V}$ onto a position in an output space $\mathcal{A}$ using a set of associated (to $\mathcal{A}$) reference vectors. To achieve this projection, vector quantizers are trained by unsupervised learning schemes. The intimate relationship between neural maps and vector quantization is manifest in the identical projection rule: map a data point to the nearest codebook vector.

In more detail, general vector quantization is an unsupervised method for data compression. This method requires that the data are given in sets of data vectors possibly of rather high dimensions. The data are approximated by a substantially smaller set $\mathbf{W}$ of reference vectors $\mathbf{w_r}$ (codebook vectors). Thereby a data vector $\mathbf{v} \in V$ is coded by this reference vectors $\mathbf{w_{s(v)}}$ the distance $||v - \mathbf{w_{s(v)}}||$ of which assumes its minimum taken over all elements of $\mathbf{W}$. Let now $\mathcal{A}$ be the index set of $\mathbf{W}$, i.e. we have a unique mapping between $\mathbf{W}$ and $\mathcal{A}$. Then we can take the coding procedure as a mapping

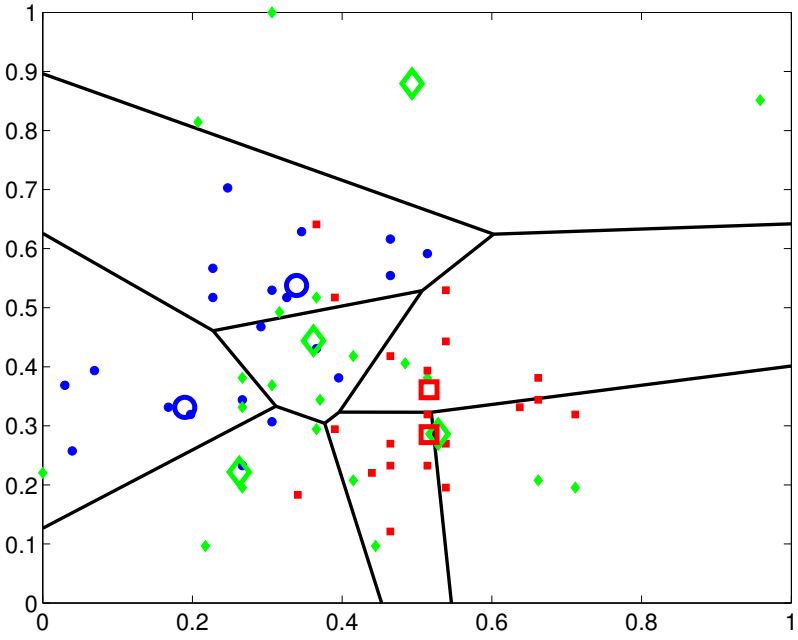


Figure 3.2: Voronoi cells for a three class data set using euclidean metric

from the input space V onto $\mathcal{A}$ in the following way:

$$\Psi_{V\to\mathcal{A}} : \mathbf{v} \mapsto \mathbf{s}\left(\mathbf{v}\right) = \mathbf{argmin}_{\mathbf{r}\in\mathcal{A}} d\left(\mathbf{v}, \mathbf{w_r}\right) \tag{3.1}$$

with

$$d\left(\mathbf{v}, \mathbf{w}\right) = ||v - \mathbf{w_r}|| \tag{3.2}$$

is based on an appropriate norm. Usually the Euclidean norm is chosen. The mapping $\Psi_{V\to\mathcal{A}}$ realizes a winner take all rule.

The neuron $\mathbf{s}$ is called winner or best matching unit. The subset of the input space

$$\tilde{\Omega_\mathbf{r}} = \left\{\mathbf{v} \in \mathbb{R}^{\mathbb{D}_\mathbf{v}} : \Psi_{V\to A}\left(\mathbf{v}\right) = \mathbf{r}\right\} \tag{3.3}$$

which is mapped to a particular neuron $\mathbf{r}$ according to (3.1), forms its Voronoi cell whereas the masked Voronoi cell is defined as $\Omega_\mathbf{r} = \tilde{\Omega}_\mathbf{r} \cap V$. According to the definition of mapping function $\Psi_{V\to A}\left(\mathbf{v}\right)$ both $\tilde{\Omega}_\mathbf{r}$ and $\Omega_\mathbf{r}$ are closed sets.

The induced Voronoi diagram V_H of a subset $H \subseteq \mathbb{R}^{\mathbb{D}_\mathbf{v}}$ with respect to a set $S = \{\mathbf{w}_i \in H \subseteq \mathbb{R}^{\mathbb{D}_\mathbf{v}}; i = 1, \ldots, n\}$ of points, is given by the set of all masked Voronoi polyhedra Ω_i. All Voronoi polyhedra perform a partitioning of H, i.e. $H = \cup_{i=1,\ldots,n}\Omega_i$. as depicted in Figure 3.2.

The crucial point in vector quantization is how one can find a good codebook W. From mathematical point of view an appropriate criterion is the expectration value of the squared reconstruction error:

$$E[\mathbf{W}] = \int ||v - \mathbf{w}_s||^2 \mathcal{P}(\mathbf{v}) d\mathbf{v}$$

where $\mathcal{P}(\mathbf{v})$ is the probability density of the data distribution of the data vectors. The simplest method to minimize the average reconstruction error $E[\mathbf{W}]$ is a stochastic gradient descent starting with initial values $\mathbf{w}_r(0)$ and updating according to

$$\mathbf{w}_r(t+1) = \mathbf{w}_r(t) - \frac{\epsilon}{2} \cdot \frac{\partial E}{\partial \mathbf{w}_r}$$

For sufficiently small ϵ the algorithm converges to a *local* minimum. The algorithm is known as LBG-algorithm named after LINDE, BUZO AND GRAY which published this method first [92]. However, in the update the knowledge of the generally unknown data density $\mathcal{P}(\mathbf{v})$ is supposed. We avoid this problem by an approximation of the integration:

$$\mathbf{w}_{s(v)}(t+1) = \mathbf{w}_{s(v)}(t) + \epsilon \cdot (\mathbf{v} - \mathbf{w}_{s(v)}(t))$$

which gives the basic learning scheme of vector quantization. As a consequence the weight vectors are shifted into the center of the gravity of their receptive fields:

$$\mathbf{w}_r = \int_{\Omega_r} \mathbf{v}\mathcal{P}(\mathbf{v})d\mathbf{v}$$

Most of the advanced algorithms are derived from this general approach.

3.3 Data analysis by SOM

The self-organizing map (SOM) constitutes one of the most popular data mining and visualization methods, mapping a given possibly highdimensional data set nonlinearly onto a low-dimensional regular lattice in a topology-preserving fashion [83]. It is sometimes also considered as a nonlinear PCA [117] allowing, under certain conditions, a large flexibility in analyzing potentially nonlinear data.

The SOM is a method for unsupervised learning, based on a grid of artificial neurons whose weights are adapted to match input vectors in a training set. It was first described by Kohonen [83] and is thus sometimes referred to as a Kohonen map. SOM is one of the most popular neural computation methods in use, and several thousand scientific articles have been written about it. Self organizing maps are widely used to visualize high dimensional data onto a lower dimensional planar grid (typically 2 or 3 dimensional rectangular or hexagonal grid). The SOM algorithm is fed with feature vectors, which can be of any dimension. In most applications, however, the number of dimensions will be high. Output maps can also be made in different dimensions: 1-dimensional, 2-dimensional, etc., but most popular are 2D and 3D maps, because of easy visual representation.

The algorithm is explained most easily in terms of a set of artificial neurons, each having its own physical location on the output map, which take part in a winner-take-all process (a competitive network) where a node with its weight vector closest to the vector of inputs is declared the winner and its weights are adjusted making them

closer to the input vector. Each node has a set of neighbours. When this node wins a competition, the neighbours' weights are also changed. They are not changed as much though. The further the neighbour is from the winner, the smaller its weight change. This process is then repeated for each input vector, over and over, for a (usually large) number of cycles. Different inputs produce different winners.

The network winds up associating output nodes with groups or patterns in the input data set. If these patterns can be named, the names can be attached to the associated nodes in the trained net. The SOM can be viewed as a constrained version of k-means clustering [66], in which the prototypes are encouraged to lie in a one-or two dimensional manifold in the feature space. The obtained projection is also referred as a constrained topological map, since the high-dimensional observations can be mapped down onto the two-dimensional coordiante systems such that data points which are neighbored in the high dimensional feature space may also lie neighbored in the planar projection with a specific topographic error. An exact mathematical definition of the topopgraphic mapping is given in [163]. Successful tools for assessing this map property are the topographic function and the topographic product [163],[6].

We now briefly review the basic notations for the SOM as given in [83].

In SOMs, usually, the neural lattice $\mathcal{A}$ is a prespecified D_A-dimensional rectangular grid of N neurons $\mathbf{r}$ which can in principle be of any dimensionality, or can extend to any dimension along its individual directions. This can be cast in a formal way by writing the output space positions as $\mathbf{r} = (i_1, i_2, i_3, \ldots), 1 < i_k < n_j$ with $N = n_1 \times n_2 \times \ldots$ denotes the overall number of nodes. Associated to each of the neurons $\mathbf{r} \in A$, is a weight vector $\mathbf{w_r}$ which determines a particular position in V, see Figure 3.3.

According to the mapping rule 3.1 of usual vector quantization we have now a mapping $\Psi_{V \to \mathcal{A}}$ from the input space V onto the neural lattice $\mathcal{A}$ whereas the back mapping is defined as

$$\Psi_{\mathcal{A} \to V} : \mathbf{r} \to \mathbf{w_r} \tag{3.4}$$

Both functions determine the neural map

$$\mathcal{M} = (\Psi_{V \to \mathcal{A}}, \Psi_{\mathcal{A} \to V}) \tag{3.5}$$

realized by the SOM network. To achieve the map $\mathcal{M}$, SOMs adapt the pointer positions $\mathbf{w_r}$ such that the input space is mapped onto the output space in a most faithful way. A sequence of data points $\mathbf{v} \in V \subseteq \mathbb{R}^{D_V}$ is presented to the map according to the data distribution density $\mathcal{P}(V)$. However, in difference to the usual vector quantization not only the winning weight vector is updated. In general, all weights go under adaptation. Thereby, in SOM cooperativeness is oriented according to the topological relations of the neurons in the grid $\mathcal{A}$. In particular, like in usual vector quantization first the current most proximate neuron $\mathbf{s}$ is determined, and then the pointer $\mathbf{w_s}$ as well as the pointer $\mathbf{w_r}$ of neurons in the topological neigborhood of $\mathbf{s}$ are also shifted towards $\mathbf{v}$ but with decreasing stength for increasing topological distance in $\mathcal{A}$

$$\delta \mathbf{w_r} = \epsilon h_{rs} (\mathbf{v} - \mathbf{w_r}). \tag{3.6}$$

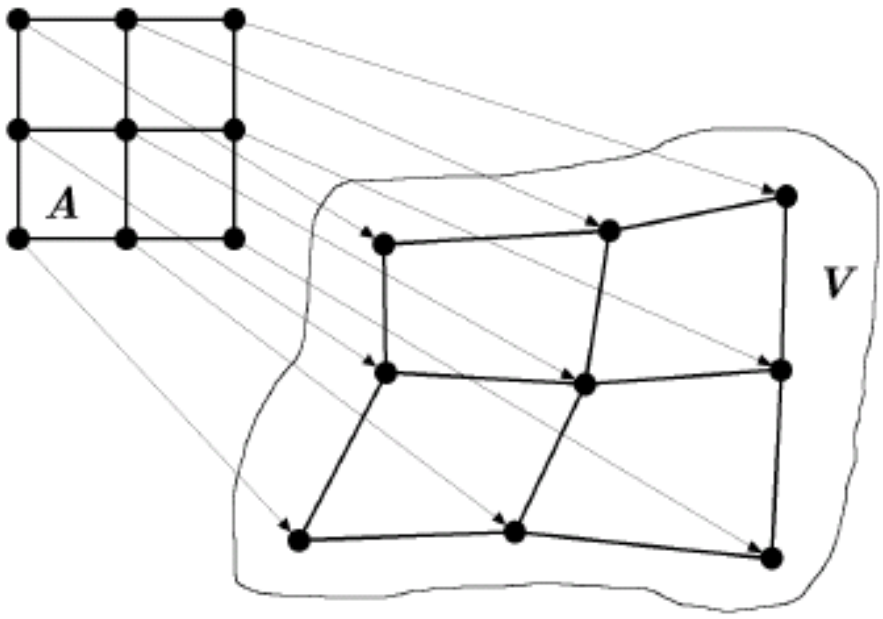


Figure 3.3: Schematic representation of the mapping of a data space onto a rectangular SOM.

In the context of neural maps the adaptation process is called learning. The parameter ϵ in 3.6 denotes the (global) learning factor, and

$$h_{rs} = exp\left(-\frac{||\mathbf{r}-\mathbf{s}||^2}{2\sigma^2}\right) \tag{3.7}$$

is the neighborhood function with neighborhood range parameter σ. The positions $\mathbf{r}, \mathbf{s}$ are used in the $\mathbb{R}^{D_V}$ dimensional space and $||\ldots||$ denotes the usual Euclidean norm. We remark that in SOMs h_{rs} is evaluated in the output space $\mathcal{A}$. It is usually chosen to be of Gaussian shape. A good introduction to SOMs is given in [83, 66].

Within the analysis of proteomic data originating from MS measurements the SOM typically yields to an initial view of the high dimensional data. This can also be used to observe cluster boundaries [149] which is helpful to determine a class labeling for variables with unsure ranges (e.g. body mass index, blood presure) depending on e.g. regional specified clinical regulations. Recent approaches also incorporate label information into the projective mapping such that the SOM has now evolved to a methodology at the borderline of supervised and unsupervised data analysis [141, 126]

A further unsupervised approach to determine a compact representation of a larger potentially high dimensional data space is given by the Neural Gas network as introduced in [94].

3.4 Neural GAS

The Neural Gas algorithm is a type of vector quantizer. In contrast to SOM, no topology of a fixed dimensionality is imposed on the network further the method gives a compact representation of the underlying data distributions.

The original Neural GAS algorithm was developed by Martinetz [94]. Assume data points $\mathbf{v} \in V$ are distributed according to $\mathcal{P}$, the goal of NG is to find prototype locations $\mathbf{w}_i \in \mathbf{W}, i = 1, \ldots, n$ such that these prototypes represent the distribution $\mathcal{P}$ as accurately as possible, minimizing the cost function:

$$Cost_{NG}(\gamma) = \frac{1}{C(\gamma, K)} \sum_{i=1}^{N} \int \mathcal{P}(\mathbf{v}) \cdot h_\gamma(i, \mathbf{v}, \mathbf{W}) \cdot (\mathbf{v} - \mathbf{w}_i)^2 \; d\mathbf{v} \tag{3.8}$$

with neighborhood function of Gaussian shape:

$$h_\gamma(i, \mathbf{v}, \mathbf{W}) = \exp\left(-\frac{k_i(\mathbf{v}, \mathbf{W})}{\gamma}\right). \tag{3.9}$$

Thereby $k_i(\mathbf{v}, \mathbf{W})$ yields the number of prototypes $\mathbf{w}_j$ for which the relation $d(\mathbf{v}, \mathbf{w}_j) \leq d(\mathbf{v}, \mathbf{w}_i)$ is valid, i.e. $k_i(\mathbf{v}, \mathbf{W})$ is the winner rank [96]. $C(\gamma, K)$ is a normalization constant depending on the neighborhood range γ and the cardinality K of $\mathbf{W}$. The NG learning rule reads as

$$\triangle \mathbf{w}_i = \epsilon \cdot h_\gamma(i, \mathbf{v}, \mathbf{W}) \cdot (\mathbf{v} - \mathbf{w}_i) \tag{3.10}$$

minimizing the cost function. The initialization of the prototypes is not longer crucial in NG because of the involved neighborhood cooperation.

Thereby, the neighborhood range γ is decreased during training to ensure independence of initialization and optimization of the quantization error. NG is a simple and highly effective algorithm for data clustering. Especially in an initial step of data analysis were a strict class labeling or the number of classes is still not available the NG algorithm can be useful for a first analysis prior further steps.

If training data $\mathbf{v_1}, \ldots, \mathbf{v_p}$ are given priorly, a fast alternative batch training schemes exist for NG called batch NG [31]. Often batch training converges after only few $(10 - 100)$ cycles such that this training mode offers considerable speedup in comparison to the online variants.

3.5 Estimation of the intrinsic dimension

, Data originating from machine measurements often are high-dimensional due to the complexity and power of the measurement technology. However in most cases the problem itself is low dimensional. For example in [170] it could be shown that a very high-dimensional problem is in fact sufficiently explained in only three dimensions or

with appropriate coding even in two dimensions. A dimensionality estimation gives us a first impression which number of relevant dimensions can be expected and hence how subsequently algorithms may be parametrized or interpreted. Different methods for dimensionality estimation has been proposed [50, 18, 167, 22] with an overview given in [20]. In this contribution the Grassberger-Proccacia dimensionality estimation (GPDim) procedure will be used which has been shown to give valid results at a reliable effort [167]. In the following we will give a very short introduction to the method of Grassberger-Proccacia and apply it to the considered analysis problems.

In nonlinear dynamics, estimation methods for the dimension of strange attractors, data sets with fractal dimension, have been a major area of research. The predominantly used procedure is the Grassberger-Proccacia-method, which yields the dimension estimate as the scaling exponent relating a correlation integral to the underlying length scale [51]. It takes only local geometrical properties of the data into account and, hence, it may be applied to estimate the data dimension of a data set [176]. Moreover, nonlinear data sets can be considered because of the local character of the measuring. Even though in nonlinear dynamics, due to the self-similar structure of strange attractors, the dimension estimate is independent of the length scale involved in the analysis (in a certain regime of length scales), this does not necessarily need to be true for the data sets investigated here. However, several applications have shown that this approach gives consistent results in comparison to other tools [161, 159, 160, 169].

Generally, the intrinsic or effective dimension D_{eff} is mathematically defined as the Hausdorff dimension $D_{\text{Hausdorff}}$.

Definition 1 (Hausdorff-measure)
Let $V \subset R^{D_v}$ a set and $C_\epsilon = \{C_i(\epsilon)\}_{i=1}^N$ a system of sets such that $V \subset C_\epsilon = \cup_{i=1}^N C_i(\epsilon)$ and $\sup_{x,y \in C_i(\epsilon)} ||x-y|| = \delta_i < \epsilon$ for each i. Let be further $\alpha(d) = \frac{\tau(\frac{1}{2})^d}{\tau(d+\frac{1}{2})}$ the volume of the d-dimensional unit ball. Then

$$H(d,V) = \lim_{\epsilon \to 0} \left[\inf_{C_\epsilon} \left(\alpha(d) \cdot \sum_i (\delta_i)^d \right) \right]$$

is called Hausdorff-dimension $D_{\text{Hausdorff}}$ of V if the limit exists, i.e. in this case we have $D_{\text{Hausdorff}} = H(d,V)$.

The Hausdorff-dimension $D_{\text{Hausdorff}}$ can be determined by use of the Grassberger-Proccacia-Approach as introduced in [51]. For its determination we have to cover the data base V by D_V-dimensional hypercubes of length ϵ and consider the minimal needed number. Let us consider for each $\mathbf{v}_i \in V$

$$p_\epsilon = \lim_{N \to \infty} \frac{1}{N} \sum_{j=1}^{N} \Theta(\epsilon - ||\mathbf{v}_i - \mathbf{v}_j||)$$

with Θ as Heavyside-function counting the number of elements $\mathbf{v}_j \in V$ contained in a

ball of radius ϵ around $\mathbf{v}_i$. One can introduce the generalized correlation integral

$$C_q(\epsilon) = \left(\frac{1}{N} \sum_{i=1}^{N} (p_\epsilon(\mathbf{v}_i))^{q-1} \right)^{\frac{1}{q-1}}$$

Then one can write the generalized Hausdorff dimension D_q as

$$D_q = \lim_{\epsilon \to 0} \frac{\ln(C_q(\epsilon))}{\ln(\epsilon)}$$

In particular one has for the correlation dimension D_2

$$D_2 = \lim_{\epsilon \to 0} \frac{\ln(C(\epsilon))}{\ln(\epsilon)}$$

with

$$C(\epsilon) = \lim_{N \to \infty} \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1, i \neq j}^{N} \Theta(\epsilon - ||\mathbf{v}_i - \mathbf{v}_j||)$$

Hence, one can derive the correlation dimension from a twice-logarithmic plot of $C(\epsilon)$. The value D_{GP} is an estimation for $D_{\text{Hausdorff}}$ and, in generally, one has $D_{\text{Hausdorff}} \leq D_{GP}$ [51, 109]. If the GPDim is sufficiently small (e.g. less than 10 dimensions) a lower dimensional visualization of the data, e.g. by a SOM, can be tried.

3.6 Supervised classification

Considering a classification problem of data, with given labels an unsupervised approach such as clustering coupled by e.g. a post labeling approach may not any longer desirable. This brings us to methods which make direct usage of the known label information and which form the concept of supervised learning.

Most often the aim of pattern analysis is to predict one feature of the data as a function of the other feature values. It is therefore to be expected that many pattern analysis tasks isolate one feature that it is their intention to predict. Hence the training data comes in the form of

$$(\mathbf{x}, y)$$

where y is the value of the feature that the system aims to predict and $\mathbf{x}$ is a vector containing the remaining feature values. The vector $\mathbf{x}$ is known as input, while y is referred to as the target output or label.

Supervised data analysis have this type of representation. For this type of task a pattern is sought in the form

$$f(\mathbf{x}, y) = \mathcal{L}(y, g(\mathbf{x}))$$

where g is referred to as the prediction function and $\mathcal{L}$ is known as a loss function. Since it measures the discrepancy between the output of the prediction function and the correct value y, we may expect the loss to be close to zero when a pattern is detected. When new data is presented the target output is not available and the pattern function is used to predict the value of y for the given input $\mathbf{x}$ using the function $g(\mathbf{x})$. Within supervised learning one tries to determine this unknown prediction function g with small loss.

In the following we will understand supervised learning as classification. Thereby we would like to get a model upon a given set of data, which is able to discriminate between at least two classes of disjunct items, giving a correct prediction of the class membership for new items matched against this model. Different approaches to model classifiers from a given set of proteomic training data have been proposed ([110, 78, 175]).

In general this problem may be seen as a probabilistic problem. Hence it is natural to look for such a probabilistic solution in terms of the Bayes model. Which says, that we classify to the most probable class, using the conditional (discrete) distribution $P(Y|X)$ with Y being a random categorial output variable and X being a random input variable for our classifier system.

3.6.1 Bayes classification

Let π_c denote the prior probability of a class c, and $p(\mathbf{v}|c)$ the density of the observations $\mathbf{v} \in V$ with $V \subseteq \mathbb{R}^{D_V}$ for the class. Then the posterior distribution of the class after observing $\mathbf{v}$ is:

$$p(c|\mathbf{v}) = \frac{\pi_c p(\mathbf{v}|c)}{p(\mathbf{v})}$$

It is fairly simple to show that the allocation rule which makes the smallest expected number of error chooses the class with maximal $p(c|\mathbf{v})$; this is known the Bayes rule.

The Bayes approach is convenient. However, the necessary estimations and the knowledge about the distribution of the data is in general not available. Therefore different alternative approaches have been proposed. We will not review the whole set of possible methods but only refer to some references on demand and present the concept of discriminant analysis in acc. to Fisher [44], which is also helpful for the subsequently derived methods.

3.6.2 Linear and Quadratic discriminant

Linear discriminant analysis (LDA) is used in machine learning to find the linear combination of features which best separate two or more classes of object. LDA is closely related to ANOVA and regression analysis, which also attempt to express one dependent variable as a linear combination of other features or measurements [66]. In the other two methods however, the dependent variable is a numerical quantity, while for

LDA it is a categorical variable (i.e. the class label). LDA is also closely related to principal component analysis (PCA) and factor analysis in that both look for linear combinations of variables which best explain the data. LDA explicitly attempts to model the difference between the classes of data. PCA on the other hand does not take into account any difference in class, and factor analysis builds the feature combinations based on differences rather than similarities.

Suppose we have a set of $N_{\mathcal{L}}$ classes, and for each case $\mathbf{v} \in V$, we know the corresponding class c. Then we can use the class information to reveal the structure of the data. Let W denote the within-class covariance matrix, that is the covariance matrix of the variables centered on the class mean, and B denote the between-classes covariance matrix, that is, of the predictions by the class means. Let M be the $N_{\mathcal{L}} \times D_V$ matrix of class means, and G be the $N_S \times N_{\mathcal{L}}$ matrix of class indicator variables (such that $G_{ij} = 1$ if and only if case i is assigned to class j). Let $\bar{x}$ be the means of the variables over the whole sample. The sample covariance matrices are:

$$W = \frac{(V - GM)^\top (V - GM)}{N_S - N_{\mathcal{L}}}, \quad B = \frac{(GM - I\bar{x})^\top (GM - I\bar{x})}{N_{\mathcal{L}} - 1}$$

with I the unity matrix. Fisher introduced a linear discriminant analysis seeking a linear combination $\mathbf{v} \cdot \mathbf{a}$ of the variables that has a maximal ratio of the separation of the class means to the within class variance, thereby maximizing the class separability. That is, maximizing the ratio $J(\mathbf{a}) = \frac{\mathbf{a}^\top B \mathbf{a}}{\mathbf{a}^\top W \mathbf{a}}$. To compute this, a scaling of the variables is chosen such that they have the identity as their within-group correlation matrix W. On the rescaled variables the problem is:

$$\begin{aligned} &\text{maximize } \mathbf{a}^\top B \mathbf{a} \\ &\text{subject to } ||\mathbf{a}|| = 1. \end{aligned}$$

The direction $\mathbf{a}$ maximizing $J(\mathbf{a})$ can be found for two classes to be $\mathbf{a} = W^{-1}(\mu_1 - \mu_2)$ as shown in [39] (p. 152ff). More general the problem is solved by taking $\mathbf{a}$ to be the eigenvector of B corresponding to the largest eigenvalue. This is the direction with the largest between class variance and the orthogonal plane separates the two classes. The linear combination $\mathbf{a}$ is unique up to a change of signs [39].

As for principal components, we can take further linear components corresponding to the next largest eigenvalues. Note that the eigenvalues are the proportions of the between-classes variance explained by the linear combinations, which may help us to choose how many to use. The corresponding transformed variables are called the linear discriminants or canonical variables. The linear discriminants are conventionally centered to have mean zero on the dataset.

The class means and covariances are in general not known. They can, however, be estimated from the training set using e.g. the maximum likelihood estimate in place of the exact value in the above equations. Although the estimates of the covariance may be considered optimal in some sense, this does not mean that the resulting discriminant

obtained by substituting these values is optimal in any sense, even if the assumption of normally distributed classes is correct.

An alternative approach to discrimination is via probability models. Let π_c denote the prior probabilities of the classes and $p(\mathbf{x}|c)$ the densities of distributions of the observations for each class. Then the posterior distribution of the classes after observing $\mathbf{x}$ is

$$p(c|\mathbf{x}) = \frac{p(\mathbf{x}|c)\pi_c}{\sum_i^{N_\mathcal{C}} p(\mathbf{x}|c_i)p(c_i)}$$

Let $p(\mathbf{x}|c)$ be a multivariate Gaussian with the special case, that the classes have a common covariance matrix. In comparison of two classes l and k, it is sufficient to look at the log-ratio, and one gets [66]

$$\log \frac{p(c=k|\mathbf{x})}{p(c=l|\mathbf{x})} = \log \frac{p(\mathbf{x}|k)}{p(\mathbf{x}|l)} + \log \frac{\pi_k}{\pi_l}$$

After some further algrebaic manipulations we get the linear discriminant functions as [66]:

$$\delta_c(\mathbf{x}) = \mathbf{x}^\top \Sigma^{-1} \mu_c - \frac{1}{2} \mu_c^\top \Sigma^{-1} \mu_c + \log \pi_c$$

with the sample covariance estimate as $\Sigma = W$, $\pi_c = N_c/N_S$, $\mu_c = \sum_{\mathbf{x_i} \in V_c} \mathbf{x_i}/N_c$. The classifier becomes

$$f(\mathbf{x}) = argmax_c \delta_c(\mathbf{x})$$

Now suppose the distribution for class c is multivariate normal with mean μ_c and covariance Σ_c. Then the Bayes rule minimizes:

$$\begin{aligned} Q_c &= -2 \log p(x|c) - 2 \log \pi_c & (3.11) \\ &= (x - \mu_c)\Sigma_c^- 1 (x - \mu_c)^\top + \log |\Sigma_c| - 2log\pi_c & (3.12) \end{aligned}$$

The first term of 3.12 is the squared Mahalanobis distance to the class center. The difference between the Q_c for the e.g. two classes is a quadratic function of $\mathbf{v}$, so the method is known as quadratic discriminant analysis (QDA) and the boundaries of the decision regions are quadratic surfaces in $\mathbf{v}$ space [39]. A precise estimation may be hard if the number of data points is small or the dimensionality of the problem becomes high.

Therefore different alternative classification approaches have been proposed which try to avoid e.g. distribution assumptions.

3.6.3 Support Vector Machines

Support Vector Machines (SVM) as introduced by Vapnik [156] constitute powerful learning machines which are based on the principle of Structural Risk Minimization (SRM). They have become quite popular in a large range of applications involving classification or regression tasks. The idea behind SVM is to map the input vectors

$\mathbf{x}$ into a high-dimensional feature space Z through some potential nonlinear mapping, chosen *a priori*. In this space, an optimal separating hyperplane is constructed [154]. The determination of the hyerplane can be defined as a quadratic optimization problem as explained in detail in [154]. We will not go into details here but only show the raw principle using a linear SVM classifier. The construction of a linear classifier for separating two classes of data in a D_V dimensional space may be equivalently viewed as to find a linear projection $\mathbb{R}^{D_V} \rightarrow \mathbb{R}$, represented by a normalized $D_V \times 1$ unit projection vector $\mathbf{v}$, such that a predefined separability measure between the two classes of data is maximized. Considering the data V as N_S column vectors $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_{N_S}}$ and their corresponding class membership as $c_1, c_2, \ldots, c_{N_S} \in \{-1, 1\}$, the expression for finding an optimal hyperplane classifier for linearly separable data is formulated as solution of the following optimization problem: Maximize $s = b_2 - b_1$ with:

$$\begin{aligned} s_{\max} &= \min_{\forall \mathbf{x}_i \in V_1, \mathbf{x}_j \in V_2} |\mathbf{v}^{\mathbf{T}}(\mathbf{x}_i - \mathbf{x}_j)| = \max_{\forall \hat{\mathbf{v}} \in V} \{ \min_{\forall \mathbf{x}_i \in V_1, \mathbf{x}_j \in V_2} |\hat{\mathbf{v}}^{\mathbf{T}}(\mathbf{x}_i - \mathbf{x}_j)| \} \\ b_{1,2} &\in \mathbb{R} \text{ bias terms} \end{aligned}$$

subject to

$$\begin{aligned} \mathbf{v} \cdot \mathbf{x_i} &\leq b_1, \text{ if } c_i = -1 \\ \mathbf{v} \cdot \mathbf{x_i} &\geq b_2, \text{ if } c_i = 1 \end{aligned}$$

where $||\mathbf{v}|| = 1, b_1 < b_2$ and b_1, b_2 are two scalars in the one dimensional projection space. Define $b = -(b_1 + b_2)/(b_2 - b_1)$ and $\mathbf{v}' = 2\mathbf{v}/(b_2 - b_1)$. We minimize $||\mathbf{v}'||$ under the restrictions:

$$\begin{aligned} \mathbf{v}' \cdot \mathbf{x}_i + b \leq -1, &\quad \text{if } c_i = -1 \\ \mathbf{v}' \cdot \mathbf{x}_i + b \geq 1, &\quad \text{if } c_i = 1 \end{aligned}$$

which can be equivalently expressed as a minimization of $\frac{1}{2}\mathbf{v}' \cdot \mathbf{v}$ subject to:

$$c_i(\mathbf{v}' \cdot \mathbf{x}_i + b) \geq 1, i = 1, 2, \ldots, N_S \tag{3.13}$$

Solving (3.13), we obtain $s_{\max} = 2/||b||$, and the $D_V \times 1$ unit projection vector $\mathbf{v} = \mathbf{v}'/||\mathbf{v}'||$. Furthermore, the vector $\mathbf{v}$ can be considered as relevance weights comparable to the relevance factors λ in GRLVQ.

To accommodate situations where two classes are not completely separable and to deal with noisy or mislabelled data, the constraints are relaxed by introducing non-negative slack variables $\xi_1, \ldots, \xi_{N_S}$ that represent errors in the constraints. These errors are penalized in the objective function:

$$\text{Minimize} \quad : \quad \frac{1}{2}\mathbf{v}' \cdot \mathbf{v} + \sum_{i=1}^{m} p_i \xi_i \tag{3.14}$$

$$\text{Subject to} \quad : \quad c_i(\mathbf{v}' \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, i = 1, 2, \ldots, N_S \tag{3.15}$$

where coefficients p_i are positive constraints representing the relative importance of individual data points. The respective dual problem is as common using the Lagrangian

multipliers α_i:

$$Q(\alpha) \stackrel{Max.!}{=} -\frac{1}{2}\sum_{i,j=1}^{N_S}\alpha_i\alpha_j c_i c_j \mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^{N_S}\alpha_i \tag{3.16}$$

with constraints

$$\sum_{i=1}^{N_S}\alpha_i c_i = 0 \text{ and } 0 \leq \alpha_i \leq p_i$$

It should be noted that if $p_i = C$ are set to be a constant C for all data points, then equation (3.14) has the same form and solution as the soft margin hyperplane linear classifier described with SVM [153]. The equation 3.16 can be easily extended to the nonlinear case substituting the inner product $\mathbf{x}_i\mathbf{x}_j$ by a non-linear kernel function $K(\mathbf{x}_i\mathbf{x}_j)$ which fits the constraints to kernel functions as defined in [156]. For the linear case the kernel may be just the ordinary inner product $K(\mathbf{x}_i\mathbf{x}_j) = \mathbf{x}_i^\top \cdot \mathbf{x}_j$. The unique solution of this optimization problem can be taken as a classifier. While the usage of a kernel gives a lot of freedom it is also a source of complexity. Using a non-linear kernel such as e.g. a *Radial Basis Function* (RBF) kernel [156] the SVM is not any longer transparent in the decision process. SVMs are very powerful formal established methods but act more as black box systems if one leaves the most simple case of two linear separable distributions [88, 87]. If appropriate SVMs will be used for comparison.

In the following we focus on prototype based models. They do not rely on specific data distribution assumption and has found to give reliable results also for small but high-dimensional data analysis problems [125, 172, 64]. Beside this fact they are quite intuitive and give a natural modeling of multi class problems which often occur in clinical applications. In that way they are often more convenient than e.g. Support Vector Machines. There are still further different approaches which can be applied to model classifiers upon proteomic data but considering the clinical data requirement as e.g. non-blackbox, confidence/fuzziness, adaptive learning for class label and identification of relevant expressed peptides, a prototype based approach appears to be more than appropriate as it will be shown in the following chapters.

3.7 Evaluation methods

Classification methods focusing on clinical problems commonly have additional demands. A clinical diagnosis system which e.g. may predict the stage or type of a disease upon an underlying classification model has to fit strong constraints on its generalization ability and accuracy. The obtained models has to show high generalization on new data, estimated by crossvalidation methods and for clinical two class approaches two terms are of special important, namely *sensitivity* and *specificity*.

3.7.1 Cross-validation

Cross-validation is a method to estimate the prediction error of a given model. Thereby the available data are splitted into disjunct sets of training and test data. This method directly estimates the extra-sample error $E[L(Y, f(X))]$, which is the generalization error when the method $f(X)$ is applied to an independent test sample form the joint distribution of X and Y. Different realization of cross-validation have been proposed, the most prominent approaches are k-fold, random and leave-one-out (LOO) cross-validation. For each method the available data are splitted into a training and a test set. The method f is trained on the training data and the prediction error is estimated on the independent test set. This procedure is iterated multiple times, while the prediction error of the crossvalidation is given as the mean of the single prediction errors. The scenario is depicted in Figure 3.4 Within a k-fold crossvalidation the available

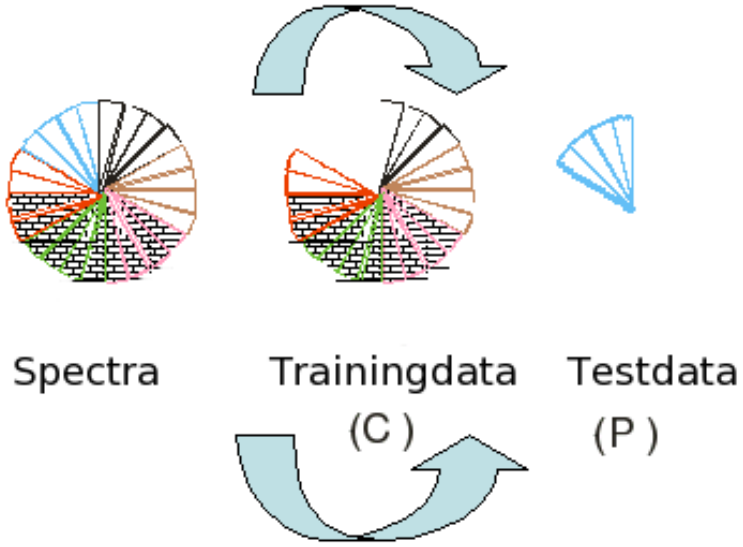


Figure 3.4: Schematic representation of a cross-validation procedure. The left plot shows multiple slices of the data. A part of these slices constitutes the training data (middle) and the remaining data the test set (plot on the right).

data are divided into k parts thereby $k - 1$ parts are used for training and one part is used to estimate the prediction error. Common choices for k are 5 or 10. The random crossvalidation splits the data randomly into training and test set which again is iterated multiple times. The LOO cross-validation takes only one item of the data (e.g. one spectrum) a side for later tests and uses the remaining data for training. LOO is sometimes considered to be too optimistic but is often used for very small data sets. A detailed introduction to cross-validation can be found in [77] and [66].

		Expected response	
Given		P	N
System	P	TP	FP
Response	N	FN	TN

Table 3.1: Confusion matrix summarizing the four categories of prediction in binary classification

3.7.2 Sensitivity and specificity

In the following we consider the case of binary classification by use of two classes with e.g. one as the control and one as the cancer class. Taking such a classifier, its performance is frequently compared using the prediction or test set accuracy taken e.g. from a k-fold crossvalidation [77]. As however already remarked for e.g. clinical classification problems this criterion is not sufficient since the significance of the two misclassification types is also important. Consequently the system performance is indeed better described in terms of the sensitivity and specificity of the classifier system quantifying their performance for false positive and false negatives. A classified test point falls always in one of the following four categories:

1. False positive (FP) - if the classifier labels the item as positive although it is negative
2. False negative (FN) - if the classifier labels the item as negative although it is positive
3. True positive (TP) - if the classifier labels the item correctly as positive
4. True negative (TN) - if the classifier labels the item correctly as negative

Using this notation we can create a confusion matrix to summarize the performance of a learning machine (c.f. Table (3.1)): A perfect predictor would have a diagonal confusion matrix (having entries 1 only at the main diagonal from upper left to lower right and 0 otherwise). We now formally define the sensitivity and specificity of such a classifier.

Definition 5 (Sensitivity) *The sensitivity ζ of a learning machine is defined as the ratio between the number of true positive predictions TP and the number of positive instances in the test set as $\zeta = \frac{TP}{TP+FN}$. This also known as the true positive fraction (TPF).*

Definition 6 (Specificity) *The specificity ς of a learning machine is defined as the ratio between the number of true negative predictions TN and the number of negative instances in the test set as $\varsigma = \frac{TN}{TN+FP}$. This also known as the true negative fraction (TNF).*

The *accuracy* is defined as the ratio between the number of correctly classified examples versus the test set size:

$$\alpha = \frac{TP + TN}{TP + TN + FP + FN}.$$

Stressing further the above example the sensitivity determines the percentage of correctly classified diseased individuals and the specificity the percentage of correctly classified individuals without a disease.

These terms are closely related to the method of *Receiver operator characteristic* (ROC) which will be introduced briefly.

3.7.3 Receiver Operator Characteristic and Penalization

ROC analysis is a classical method in Signal Detection Theory [147] and is used also in statistics, medical diagnostics and more recently Machine Learning as an alternative method for comparing learning systems [113].

For many classifier systems it is important to distinguish the two types of errors that can arise, namely a false alarm is usually not as expensive as a missed correct alarm. This brings us to the concept of Receiver Operator Characteristic analysis (ROC) ([16]) which is a method to describe the performance of a two class classifier (dichotomizer) at different operating points; its main peculiarity is its independence of the a priori probabilities of the two classes and this makes ROC particularly effective for analyzing the dichotomizer behavior under different class and cost distributions.

For example within clinical cancer research it is important to avoid false negative results, but a small number of false positive results may be tolerated if the classifier results are re-screened by medical trained personnel or additional tests. We give a short example (taken from [177]) indicating the relevance of this problem in the clinical domain and introduce the necessary terms.

False negatives are a significant issue in medical testing. In some cases, there are two or more (often many) tests that could be used, one of which is simpler and less expensive, but less accurate, than the other. For example, the simplest tests for HIV and hepatitis in blood have a significant rate of false positives. These tests are used to screen out possible blood donors, but more expensive and more precise tests are used in medical practice, to determine whether a person is actually infected with these diseases.

False negatives in medical testing provide false, incorrect reassurance to both patients and physicians that patients are free of the disease which is actually present. This in turn leads to people receiving inappropriate understanding and a lack of better advice and treatment to better protect their interests. A common example is relying on cardiac stress tests to detect coronary atherosclerosis, even though cardiac stress tests are known to only detect limitations of coronary artery blood flow due to advanced stenosis.

Lets have a test for a disease with 99% sensitivity and 99% specificity. Say you test 2000 people, 1000 of them are sick and 1000 of them are healthy. You are likely to get about 990 true positives, 990 true negatives, and 10 false positives and 10 false negatives. The positive and negative prediction values would be 99%, so the people can be quite confident about the result.

Say, however, that of the 2000 people only 100 are really sick. Now you are likely to get 99 true positives, 1 false negative, 1881 true negatives and 19 false positives. Of the $19 + 99$ people tested positive, only 99 really have the disease - that means, intuitively, that given that your test result is positive, there's only 84% chance that you really have the disease. On the other hand, given that your test result is negative, you can really be reassured: there's only 1 chance in 1881, or 0.05% probability, that you have the disease despite of your test result.

Now, the ROC space denotes a coordinate system of a classifier, where the true positive rate is plotted on the $y-$axis and the false positive rate on the $x-$axis. In this way classifiers are compared not by a single number (e.g. the crossvalidation accuracy) but by a point in the plane. For classifiers obtained by thresholding a real valued function depending on a real parameter, this produces a curve, called a ROC curve, describing the trade-off between sensitivity and specificity. A typical scenario for ROC analysis is depicted in Figure 3.5.

Two systems can therefore be compared with the better one having the highest and leftmost points in the ROC space. Originally this method is introduced for two classes only (in our case disease and control). Recently an extension to the multiclass case has been proposed [43].

The ROC analysis brings us further to penalty control which aims on the possibility to modify the learning behavior of the considered algorithm. Thereby one is able to control which classes from the data distribution are especially well represented whereas other classes maybe less well modeled. In general an algorithm aims on modelling the whole data as well as possible typically minimizing some objective criterion e.g. an energy function. With penalty control one controls different costs for misclassification's. Especially for overlapping classes it may not be possible to obtain a perfect representation. In the prediction phase this may lead to misclassifications. By penalty control the model learning can be modified, such that underrepresented classes become better represented by accepting additional misclassifications within other classes.

This behaviour is frequently demanded within clinical applications. Now we switch to a multiclass scenario. For example we may be confronted with a clinical cancer classification problem in three classes. The first class contains data from the control group whereas the second and third class may contain data of cancer in different cancer stages. The classifier is trained to represent the data distribution in the three classes as good as possible. Here it could happen that we get misclassifications between all different classes. By penalty control one can modify the learning behavior to improve the classification performance for control vs. cancer, ignoring maybe new misclassification's between the different cancer stages as long as control and cancer are sufficiently represented.

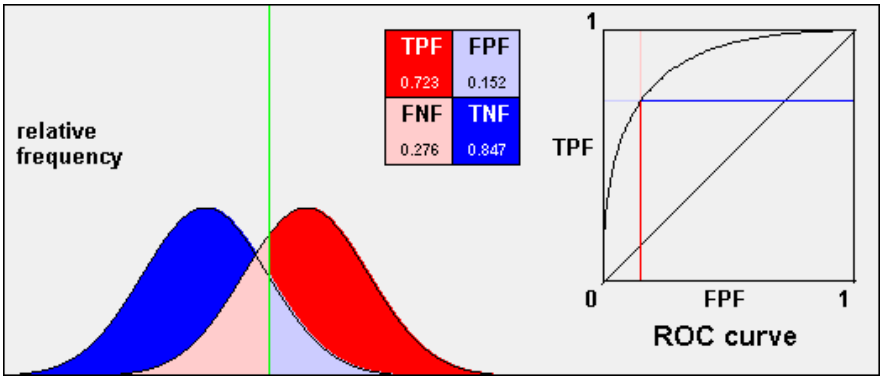


Figure 3.5: Typical scenario for a ROC analysis considering two overlapping Gaussian distributions. Thereby the (statistical) test is assumed to be *positive* if the value was above some arbitrary cutoff (threshold), and *negative* if below. Central to the idea of ROC curves is this idea of a cutoff level. Considering the two populations - diseased and non-diseased patients, the test is applied to each patient in each population in turn, and one get numeric results for each patient. Now histograms of these results, for each population are drawn. Thereby, the green line indicates the arbitrary chosen threshold. If one moves the test threshold from left to right, the proportion of false positives decreases. Unfortunately, there is a problem - as we decrease the false positives, so the true positives also decrease. A ROC curve is an exploration of what happens to TPF and FPF as we vary the position of the arbitrary TEST threshold. The point corresponding to the chosen threshold is shown on the ROC curve as the cross. If the threshold is very high, then there will be almost no false positives but one won't really identify many true positives either. Both TPF and FPF will be close to zero, so one is at a point low down and to the left of the ROC curve. If the test threshold is moved towards a more reasonable, lower value, the number of true positives will increase (rather dramatically at first, so the ROC curve moves steeply up). Finally, one reach a region where there is a remarkable increase in false positives - so the ROC curve slopes off as one moves the test threshold down to a ridiculously low value.

An additional important point especially in clinical decision systems is the unbalanced costs originating from a classification decision. Most often the decision of a classifier system assumes equal costs for incorrect classifications between different classes. In clinical scenaries however a decision for cancer or control (healthy) has typically no equal penalty. A misclassification to control (taking a diseased person as healthy) is obviously more expensive than to misclassify a healthy patient to the disease class. Accepting this fact a clinical classification systems should be able to deal with unequal costs for a classification decision also known as penalization. To incorporate penalization into a classifier system a modification of the underlying cost function is desirable.

Chapter 4

Supervised Neural Vector Quantization

The beautiful thing about learning is nobody can take it away from you.
B.B. King

The learning vector quantization algorithm (LVQ) proposed by Kohonen [83] is a prototype based supervised classifier designed to find proper data representatives in the input space. In the presence of data labels, a good prototype location must make a compromise of pointing to a possibly local center of gravity of the data belonging to the prototype's class and of well-separating data of the prototype's class from those with different labels. Typical applications of vector quantization are classification and data compression. Both features have been successfully realized by means of LVQ - or an extension of it - for image compression and image analysis [19], optical character recognition [171], speech analysis and recognition, signal waveform classifications, and robotics sensory implementations [2, 165]. Kohonen points out that the main purpose of LVQ is statistical classification or pattern recognition. Specific examples of successful applications are e.g. in medical diagnostics, molecular sequence classification in bio-informatics [125], remote sensing image analysis in geoscience [168, 169] and others.

4.1 Learning Vector Quantization

In the supervised case of class prediction for a given data point, the k-nearest neighbor algorithm (k-NN) proposed by Cover and Hart [32] is one of the earliest and still frequently used classifiers. It is a lazy learner which memorizes all samples in the training set and which predicts the class of unknown data by a soft majority vote weighted by the distances of the k nearest stored points. Kohonen's learning vector quantization (LVQ) reduces the high memory requirement of k-NN: LVQ is a sparse classifier with a small number of prototypes; thus, LVQ needs significantly less comparisons for op-

eration than k-NN, but it still leads to crisp classification results of high generality [82].

A LVQ network is determined by the prototype neurons and a distance measure on the input data, which is quite different from the paradigmatic feed-forward neural net architecture where input data are passed through error-reducing adaptive connections between neuron layers to an output layer. The LVQ dynamic that aims at improving the classification accuracy is easily obtained by paying tribute to the introduced distortion function E. For realizing supervised prototype positioning with labeled data, an extended indicator function χ can be used, for example, by choosing $\chi = 1$ for the closest prototype representing the same label as the given pattern, $\chi = -1$ for the closest prototype with a different label, and $\chi = 0$ else. This motivates a Hebbian learning approach.

Hebbian learning is a hypothesis for how neuronal connections are enforced in mammalian brains; it is also a technique for weight selection in artificial neural networks. The idea is named after Donald Hebbs book *The Organization of Behavior* [68] (and inspired research into neural networks as a result). His idea specified how much the strength of a connection between two neurons should be altered according to how they are firing at that time. Hebb's original principle was essentially that if one neuron is stimulating some other neuron repeatedly, then the strength of the connection between the two neurons will be increased.

From the point of view of artificial neurons and artificial neural networks, Hebb's principle can be described as a method of determining how to alter the weights between model neurons. The weight between two neurons will increase if the two neurons activate simultaneously; it is reduced if they activate separately. Nodes which tend to be either both positive or both negative at the same time will have strong positive weights while those which tend to be opposite will have strong negative weights.

With this choice, an intuitive Hebbian distortion minimization strategy is locally given by moving the closest correct prototype a little towards the presented pattern and by moving away the closest wrong prototype. The class membership of an unknown data point is obtained by assigning the class of the closest best-matching prototype which captures the input pattern inside its region of responsibility. The corresponding boundaries are defined by the relative positions of the prototypes within their neighborhood structure. Learning success is based on the assumption that minimizing the average distance of labeled prototypes to data points is related to maximizing the classification accuracy. However, for strongly asymmetric or discretized data this relation distortion minimization $\Rightarrow$ classification improvement does no longer hold, unless a metric more appropriate than the Euclidean is chosen.

A key advantage of LVQ is that after training, its final prototypes can be easily interpreted, because they live in the same space as the processed data: prominent modes of the data are characterized by only a few reference vectors. According to Kohonen [83], the decision borders for a two-class LVQ problem approximate the Bayesian optimal decision border. This has been proven for the assumption of Gaussian distributed data and the squared Euclidean distance, which leads to piecewise

linear class-separating planes along the receptive field boundaries.

Hebbian online learning is achieved during iterative pattern presentations by local updates of the winner neurons. Winner prototypes with correct labels are rewarded by making them more similar to the current input, and wrong prototypes are penalized by pushing them away into opposite direction of the pattern. As a result, LVQ locates the real-valued prototype vectors in the input space by interpolating local centers of gravity of data points with common label, thereby accomplishing active separation from other classes.

Several problems of the standard LVQ algorithm can be addressed to its initialization. One problem is the choice of the number of prototypes per class. Kohonen states that the optimum number is strongly depending on the data distribution, making iterative trials necessary [83]. Too few prototypes do not capture all important data clusters and too many prototypes not only lead to long processing times but also to poor generalization, because too detailed characteristics learned from the training set cannot be transferred to new data.

Secondly, the initial placement of the prototypes contributes much to the success of LVQ. It turns out to be an unfavorable strategy to use data vectors as initial prototypes, because some data clusters might be accidentally overlooked while outliers are boosted, this way accounting for suboptimal training results. Kohonen proposes two strategies for choosing the starting locations: (1) by selecting a subset of reference vectors resulting from a previously determined k-nearest neighbor classifier, vectors for which even a large k-neighborhood represents the same class, or (2) by initialization with prototypes from the unsupervised SOM, for which the neurons are turned by majority vote into labeled LVQ prototypes [83]. These two initialization strategies themselves require a number of well-chosen parameters, such as their own initial prototype positions and considerations about the number k for k-NN, or the appropriate architecture for the SOM. Alternatively, the prototypes from the robust neural gas quantizer introduced before can be labeled after convergence and then be fine-tuned by LVQ [21]. In a later chapter, this idea will lead to an integrated model of LVQ and NG.

For learning vector quantization supervised approaches are mainly influenced by the standard algorithms LVQ1...LVQ3 introduced by KOHONEN [83] as intuitive prototype-based clustering algorithms. Several derivatives were developed to improve the standard algorithms to ensure, for instance, faster convergence, a better adaptation of the receptive fields to optimum Bayesian decision, or an adaptation for complex data structures, to name just a few [63, 84, 136].

Standard LVQ does not possess a cost function in the continuous case; it is based on the heuristic to minimize misclassifications using Hebbian learning. One of the first versions of learning vector quantization based on a cost function, which formally assesses the misclassifications, is the Generalized LVQ (GLVQ) [122].

4.2 Generalized Learning Vector Quantization

We shortly summarize the notations and learning rules for GLVQ as introduced by Sato & Yamada in [122]. Let $c_{\mathbf{v}} \in \mathcal{L}$ be the label of input $\mathbf{v}$, $\mathcal{L}$ a set of labels (classes) with $\#\mathcal{L} = N_{\mathcal{L}}$. As LVQ, GLVQ uses a fixed number of prototypes for each class. Let $\mathbf{W} = \{\mathbf{w_r}\}$ be the set of all codebook vectors and $c_{\mathbf{r}}$ be the class label of $\mathbf{w_r}$. Furthermore, let $\mathbf{W}_c = \{\mathbf{w_r} | c_{\mathbf{r}} = c\}$ be the subset of prototypes assigned to class $c \in \mathcal{L}$.

The training algorithm adapts the prototypes such that for each class $c \in \mathcal{L}$, the corresponding codebook vectors $\mathbf{W}_c$ represent the class as accurately as possible. This means that the set of points in any given class $V_c = \{\mathbf{v} \in V | c_{\mathbf{v}} = c\}$, and the union $\mathcal{U}_c = \bigcup_{\mathbf{r}|_{\mathbf{w_r} \in \mathbf{W}_c}} \Omega_{\mathbf{r}}$ of receptive fields of the corresponding prototypes should differ as little as possible. Let $f(x) = (1 + \exp(-x))^{-1}$ be the logistic function. The cost function of GLVQ is defined as

$$Cost_{GLVQ} = \sum_{\mathbf{v}} f(\mu(\mathbf{v})) \tag{4.1}$$

with

$$\mu(\mathbf{v}) = \frac{d_{\mathbf{r}_+} - d_{\mathbf{r}_-}}{d_{\mathbf{r}_+} + d_{\mathbf{r}_-}}, \tag{4.2}$$

as classifier function and $d_{\mathbf{r}_+}$ is the squared distance of the input vector $\mathbf{v}$ to the nearest codebook vector labeled with $c_{\mathbf{r}_+} = c_{\mathbf{v}}$, say $\mathbf{w}_{\mathbf{r}_+}$, and $d_{\mathbf{r}_-}$ is the squared distance to the best matching prototype but labeled with $c_{\mathbf{r}_-} \neq c_{\mathbf{r}_+}$, say $\mathbf{w}_{\mathbf{r}_-}$. Note that $\mu(\mathbf{v})$, the classifier, is negative if $\mathbf{v}$ is correctly classified. As it was shown in [121], the usage of the function $\mu(\mathbf{v})$ yields a more robust behavior in particular for overlapping classes.

The learning rule of GLVQ is obtained taking the derivatives of the above cost function performing a gradient descent. Using $\frac{\partial \mu(\mathbf{v})}{\partial \mathbf{w}_{\mathbf{r}+}} = \xi^+ \frac{\partial d_{\mathbf{r}_+}}{\partial \mathbf{w}_{\mathbf{r}+}}$ and $\frac{\partial \mu(\mathbf{v})}{\partial \mathbf{w}_{\mathbf{r}-}} = \xi^- \frac{\partial d_{\mathbf{r}-}}{\partial \mathbf{w}_{\mathbf{r}-}}$ with

$$\xi^+ = \frac{2 \cdot d_{\mathbf{r}_-}}{(d_{\mathbf{r}_+} + d_{\mathbf{r}_-})^2} \text{ and } \xi^- = \frac{2 \cdot d_{\mathbf{r}+}}{(d_{\mathbf{r}_+} + d_{\mathbf{r}_-})^2} \tag{4.3}$$

one obtains for the weight updates [122]:

$$\triangle \mathbf{w}_{\mathbf{r}_+} = \epsilon^+ \cdot f'|_{\mu(\mathbf{v})} \cdot \xi^+ \cdot \frac{\partial d_{\mathbf{r}_+}}{\partial \mathbf{w}_{\mathbf{r}+}} \tag{4.4}$$

$$\triangle \mathbf{w}_{\mathbf{r}_-} = -\epsilon^- \cdot f'|_{\mu(\mathbf{v})} \cdot \xi^- \cdot \frac{\partial d_{\mathbf{r}-}}{\partial \mathbf{w}_{\mathbf{r}-}} \tag{4.5}$$

ϵ^+, ϵ^- are learning rates. As shown in [123], the above learning rules are also valid in case of a continuous data distribution.

4.2.1 Hypothesis margin

Margin analysis of an algorithm is important to asses the level of confidence of a classifier with respect to its decision. For example, the *sample margin* is defined as the

distance between the input and the decision boundary. The natural choice of this margin to be maximized in learning vector quantization causes numerical instabilities [33]. An alternative definition gives the *hypothesis margin*: this margin is the distance that the classifier can move without changing the way it labels any sample data. Thereby, the sample-margin majorizes the hypothesis margin. The hypothesis margin of a prototype based classifier is given by

$$\left(d_{\mathbf{r}_+}\right)^{\frac{1}{2}} - \left(d_{\mathbf{r}_-}\right)^{\frac{1}{2}} \tag{4.6}$$

In fact, the GLVQ maximizes a cost function closely related to this hypothesis margin. Therefore, it can be taken as maximum margin algorithm [59]. Further, any given margin provides an upper bound for the generalization error of the classifier such that the higher the margin the lower the generalization error [33]. For a more detailed analysis we refer to [33],[59].

4.2.2 Penalization for GLVQ

In the following the controlling of penalization is formally introduced into the concept of GLVQ networks by modifying the underlying cost function. The basic idea is to define penalty terms on the distances by means of the margin used in the cost function 4.1 (or derived) which gives different loss functions for positively and negatively labelled points. Thereby we keep the original update dynamic of GLVQ. In that way we increase costs for the class where the costs of misclassification are more expensive. In turn, this induces a decision boundary which is much more distant from the *critical* class than from the other - or in short the margin is larger. The usefulness of this idea is supported by results for the misclassification probability of a point with respect to its distance from the boundary [139].

To do so we modify the logistic function f in a heuristic manner. Let $\wp = C \times C$ be a suitable chosen penalty matrix with entries $[0, 1]$ for each class to class relation. We define a penalty term $\wp_{st}$ as the penalty applied to a classification of a data point with class label c_s which is classified to the class with label c_t. We assume that a penalty of 1 indicates maximal penalization whereas a value of 0 indicates no additional penalization as it will be the case for all main diagonal elements of the matrix $\wp$. We now incorporate the matrix $\wp$ in the distance calculation of the GLVQ learning process and define a penalized logistic function:

$$f_{\wp}\left(\mathbf{v}, s, t\right) = \left(1 + \exp\left(-(1 + \wp_{(s,t)})\mathbf{v}\right)\right)^{-1} \tag{4.7}$$

with $\wp_{(s,t)} \in \wp$ as the current penalization of class s with respect to class t, thereby s indicates the true class of the data point item under optimization and t the class index of the winner. The cost function 4.1 is changed to incorporate the three argument structure of $f_{\wp}$:

$$Cost_{GLVQ_{\wp}} = \sum_{\mathbf{v}} f_{\wp}(\mu(\mathbf{v}), s, t) \tag{4.8}$$

$\wp_{12}$	#**R** err_1	#**R** err_2	**R** acc.	#**P** err_1	#**P** err_2	**P** acc.	sens.	spec.
0	35	53	92.5%	15	30	92.9%	94%	90.9%
1	23	61	92.8%	14	33	92.6%	96.1%	89.5%

Table 4.1: Results of penalization for class 1. Sensitivity and Specificity are calculated using the confusion matrix of the training data.

To keep the energy function (4.1) stable the penalization values has to be limited such that updates remain in the range of the standard GLVQ update.

Considering the learning behaviour of GLVQ networks the introduction of a penalty matrix $\wp$ into the cost function results in the following semantic behavior. If the penalty term $\wp_{st} = 0$ the updates behave as within the ordinary GLVQ. The update equations are in complete analogy to GLVQ but with an appropriate modified differentiation of the sigmoid function introducing the penalty term as a new scaling value of the update.

For an update induced by a data point $\mathbf{v}_s$ we get the corresponding penalization terms $\wp_{st}$ with $t \in \{1, \dots C\}$ in row s of matrix $\wp$. If the row contains a non vanishing penalization for the closest incorrect prototype w_t^- the obtained margin fraction $\mu_\gamma(\mathbf{v})$ is weighted by this penalty term in case of misclassifications. In that way the update rule of the GLVQ network will stronger repels the corresponding prototype w_t^-. while stronger attracting the closest correct prototype w_t^+. If now a correct prototype is in range of the data point $\mathbf{v}_s$ the potential receptive field of this prototype is shifted in its direction. During the optimization process we obtain a margin shift between the interacting classes to represent the data point $\mathbf{v}_s$ with the additional constraint of the penalties.

4.2.3 Simulations

On a synthetic data set of two overlapping Gaussians as described in [131] the penalized GLVQ with one prototype per class has been analyzed. The overlapping of the Gaussians gives natural errors for both classes, hence one can penalize the classification for the one and the other class. Without penalization the two data clouds can be recognized with 92.5% accuracy, thereby one has 35 misclassifications for class 1 and 53 for class 2. If one first wants to reduce the misclassifications for class 1 by assignment of corresponding larger costs we get the results as shown in Table (4.1) and similar for class 2 as depicted in Table (4.2).

The penalization helps to improve the recognition accuracy of the chosen single class as expected while increasing the number of misclassifications for the alternative class. Considering sensitivity and specificity values one is now able to modify the classifier modeling with respect to clinical needs.

$\wp_{21}$	#**R** err_1	#**R** err_2	**R** acc.	#**P** err_1	#**P** err_2	**P** acc.	sens.	spec.
0	35	53	92.5%	15	30	92.9%	94%	90.9%
1	53	43	91.8%	28	22	92.1%	91%	92.6%

Table 4.2: Results of penalization for class 2. Sensitivity and Specificity are calculated using the confusion matrix of the training data.

4.3 Supervised Neural GAS

An improvement of GLVQ is the incorporation of neighborhood cooperativeness according to Neural Gas to avoid local minima and sensitivity to initialization [164]. It can also be seen as an extension of usual NG for supervised learning referred as Supervised Neural Gas (SNG). Let K_c denote the cardinality of $\mathbf{W}_c$. Further we assume to have m data vectors $\mathbf{v}_i$. The neighborhood learning in SNG for a given input $\mathbf{v}_i$ with label c is applied only to the subset $\mathbf{W}_c$. The SNG cost function reads as

$$Cost_{SNG}(\gamma) = \sum_{i=1}^{m} \sum_{\mathbf{w_r} \in \mathbf{W}_c} \frac{h_\gamma(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i}) \cdot f(\mu(\mathbf{v}))}{C(\gamma, K_{c_i})} \tag{4.9}$$

with $f(x) = (1 + \exp(-x))^{-1}$ and, now, μ as defined at (4.2). The neighborhood cooperativeness makes sure that prototypes are spread faithfully among data of their respective classes. Note that

$$\lim_{\gamma \to 0} Cost_{SNG}(\gamma) = Cost_{GLVQ}. \tag{4.10}$$

However, if the neighborhood range γ is large, typically at the beginning of the training, the prototypes of one class share their responsibilities for a given input. Hence, neighborhood cooperativeness is involved such that initialization of the prototypes is not longer crucial. Given a training example $(\mathbf{v}_i, c_i)$ all prototypes $\mathbf{w_r} \in \mathbf{W}_{c_i}$ and the closest wrong prototype $\mathbf{w_{r-}}$ are adapted. Taking now

$$\xi_{\mathbf{r}}^{+} = \frac{2 \cdot d_{\mathbf{r}_-}}{(d_{\mathbf{r}} + d_{\mathbf{r}_-})^2} \text{ and } \xi_{\mathbf{r}}^{-} = \frac{2 \cdot d_{\mathbf{r}}}{(d_{\mathbf{r}} + d_{\mathbf{r}_-})^2} \tag{4.11}$$

the update rules appear as

$$\triangle \mathbf{w_r} = 2\epsilon^{+} \cdot \xi_{\mathbf{r}}^{+} \cdot \frac{f'|_{\mu(\mathbf{v})} \cdot h_\gamma(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i})}{C(\gamma, K_{c_i})} \cdot \frac{\partial d_{\mathbf{r}}}{\partial \mathbf{w_r}} \tag{4.12}$$

$$\triangle \mathbf{w_{r-}} = -2\epsilon^{-} \cdot \sum_{\mathbf{w_r} \in \mathbf{W}_c} \xi_{\mathbf{r}}^{-} \cdot \frac{f'|_{\mu(\mathbf{v})} \cdot h_\gamma(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i})}{C(\gamma, K_{c_i})} \cdot \frac{\partial d_{\mathbf{r}-}}{\partial \mathbf{w_{r-}}} \tag{4.13}$$

Yet, one could also include neighborhood cooperation for wrong prototypes. However, as shown in [164], this yields instabilities in learning.

Part II

Self-adapted metric and label information

Chapter 5

Relevance Learning in Learning Vector Quantization

Learning is about more than simply acquiring new knowledge and insights; it is also crucial to unlearn old knowledge that has outlive its relevance. Thus, forgetting is probably at least as important as learning.
Gary Ryan Blair

Pattern classification plays an important role in data processing. It is used for discrimination of patterns according to certain criteria, which may be of statistical type, structural differences, feature discrimination, etc. Thereby the representation of the objects significantly influences the ability for discrimination. An improper representation of the object may lead to vanishing differences whereas a suitable representation offers a clear separation of object classes. Naturally, the similarity measure plays a crucial role for classification of high-dimensional data and an appropriate choice of the distance might face severe difficulties.

The design of general similarity measures which can be used for any learning tasks, i.e. which guarantee a universal approximation ability and distinguish-ability of arbitrary structures, is one possible line of research; however, the resulting representation of data is often too complex for the concrete task and, moreover, a universal design might not be efficient for complex data structures. Therefore, similarity measures which are constructed for the concrete problem based on the given data are particularly interesting since they offer an automated design of problem specific representations.

A prime example of this idea is the Fisher kernel which derives a similarity measure based on a statistical model of the data [71]. Still, the resulting kernel is fairly general since it mirrors general statistical properties of the given data set. An alternative problem specific kernel has been proposed in [143] which is specifically suited for the biological problem of splice site recognition.

Domain specific knowledge is often needed to define an appropriate similarity measure for the data analysis task. One prominent example is the Tanimoto distance measure which is used often in chemometrics [116]. Another measure is given by the

scaled Euclidean metric which makes use of specific knowledge about the relevance of single input dimensions. The benefits of such problem specific metrics has already been shown in multiple contributions [145, 55, 172, 146, 125].

The choice or determination of an appropriate metric is a critical issue [172]. In particular for supervised learning only a small number of input dimensions may be relevant for the classification task, which typically can be realized as a parameterized metric. The method to determine such parameters is known as relevance learning. We now give a short review of relevance learning for LVQ networks and show derivations of these methods introducing alternative metric adaptations.

As described before in 4.2 the discriminant property of LVQ networks is realized by positioning of the prototypes such that they adapt specifically according to the given classes. As it is described in more detail later, non-standard metrics as well as metric adaptation can easily be included in advanced modifications of LVQ.

5.1 Relevance Learning in GLVQ and SNG

Supervised Neural Gas is considered as a representative for prototype based classification approaches. It can be combined with the demanded feature of relevance learning. Moreover, it is a stochastic gradient descent algorithm, which is a margin optimizer with known bounds of generalization ability [60].

We now present the respective formal descriptions including a parametrized metric. Thereby the formerly introduced notations of GLVQ and SNG are adapted appropriately. Again the task of vector quantization is realized by the map Ψ as a winner-take-all rule, as before but now assuming a general parametrized similarity measure[1]. Thereby Ψ is now given as:

$$\Psi^{\lambda}_{V\to\mathcal{A}} : \mathbf{v} \mapsto \mathbf{s}\left(\mathbf{v}\right) = \mathbf{argmin}_{\mathbf{r}\in A} d^{\lambda}\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}}\right) \tag{5.1}$$

with $d^{\lambda}\left(\mathbf{v}, \mathbf{w}\right)$ being an arbitrary often differentiable distance measure depending on a parameter vector λ. For the moment we take λ as fixed.

We now consider the Generalized Relevance Learning Vector Quantization approach - GRLVQ. GRLVQ minimizes the cost function

$$Cost_{GRLVQ} = \sum_{\mathbf{v}} f(\mu_{\lambda}(\mathbf{v})) \tag{5.2}$$

$$\mu_{\lambda}(\mathbf{v}) = \frac{d^{\lambda}_{\mathbf{r}_+} - d^{\lambda}_{\mathbf{r}_-}}{d^{\lambda}_{\mathbf{r}_+} + d^{\lambda}_{\mathbf{r}_-}} \tag{5.3}$$

via stochastic gradient descent, where $d^{\lambda}_{\mathbf{r}_+}$ is the squared weighted distance of the input vector $\mathbf{v}$ to the nearest codebook vector labeled with $c_{\mathbf{r}_+} = c_{\mathbf{v}}$, say $\mathbf{w}_{\mathbf{r}_+}$, and $d^{\lambda}_{\mathbf{r}_-}$ is

[1] A similarity measure is a non-negative real-valued function of two variables, which, in contrast to a distance measure does not necessarily fulfill the triangle inequality and has not to be symmetric. Naturally, each distance measure is a similarity measure.

the squared distance to the best matching prototype but labeled with $c_{\mathbf{r}_-} \neq c_{\mathbf{v}}$, say $\mathbf{w}_{\mathbf{r}_-}$. As it was shown in [121], the usage of the function $\mu_\lambda(\mathbf{v})$ yields robust behavior whereas LVQ2.1 does not.

The learning rule of GRLVQ is obtained taking the derivatives of the above cost function. Using $\frac{\partial \mu_\lambda(\mathbf{v})}{\partial \mathbf{w}_{\mathbf{r}+}} = \xi^+ \frac{\partial d^\lambda_{\mathbf{r}_+}}{\partial \mathbf{w}_{\mathbf{r}+}}$ and $\frac{\partial \mu_\lambda(\mathbf{v})}{\partial \mathbf{w}_{\mathbf{r}-}} = \xi^- \frac{\partial d^\lambda_{\mathbf{r}_-}}{\partial \mathbf{w}_{\mathbf{r}-}}$ with

$$\xi^+ = \frac{2 \cdot d^\lambda_{\mathbf{r}_-}}{(d^\lambda_{\mathbf{r}_+} + d^\lambda_{\mathbf{r}_-})^2} \text{ and } \xi^- = \frac{2 \cdot d^\lambda_{\mathbf{r}+}}{(d^\lambda_{\mathbf{r}_+} + d^\lambda_{\mathbf{r}_-})^2} \tag{5.4}$$

one obtains for the weight updates [122]:

$$\triangle \mathbf{w}_{\mathbf{r}_+} = \epsilon^+ \cdot f'|_{\mu_\lambda(\mathbf{v})} \cdot \xi^+ \cdot \frac{\partial d^\lambda_{\mathbf{r}_+}}{\partial \mathbf{w}_{\mathbf{r}+}} \tag{5.5}$$

$$\triangle \mathbf{w}_{\mathbf{r}_-} = -\epsilon^- \cdot f'|_{\mu_\lambda(\mathbf{v})} \cdot \xi^- \cdot \frac{\partial d^\lambda_{\mathbf{r}_-}}{\partial \mathbf{w}_{\mathbf{r}-}} \tag{5.6}$$

ϵ^+, ϵ^- are learning rates. As shown in [123], the above learning rules are also valid in case of a continuous data distribution.

The respective cost function for SNG with parametrized metric referred as Supervised Relevance Neural Gas (SRNG) becomes,

$$Cost_{SRNG}(\gamma) = \sum_{i=1}^{m} \sum_{\mathbf{r}|\mathbf{w}_\mathbf{r} \in \mathbf{W}_{c_i}} \frac{h_\gamma(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i}) \cdot f(\mu_\lambda(\mathbf{r}, \mathbf{v}))}{C(\gamma, K_{c_i})} \tag{5.7}$$

with again $f(x) = (1 + \exp(-x))^{-1}$, $h_\gamma(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i})$ as (3.9) and

$$\mu_\lambda(\mathbf{r}, \mathbf{v}) = \frac{d^\lambda_\mathbf{r} - d^\lambda_{\mathbf{r}_-}}{d^\lambda_\mathbf{r} + d^\lambda_{\mathbf{r}_-}} \tag{5.8}$$

whereby $d^\lambda_{\mathbf{r}_-}$ is defined as in GRLVQ above and $d^\lambda_\mathbf{r} = d^\lambda(\mathbf{v}, \mathbf{w}_\mathbf{r})$. The neighborhood cooperativeness makes sure that prototypes are spread faithfully among data of their respective classes. Note that $\lim_{\gamma \to 0} Cost_{SNG}(\gamma) = Cost_{GLVQ}$ holds [164]. Hence, for vanishing neighborhood the SRNG also becomes GRLVQ. Given a training example $(\mathbf{v}_i, c_i)$ all prototypes $\mathbf{w}_\mathbf{r} \in \mathbf{W}_{c_i}$ and the closest wrong prototype $\mathbf{w}_{\mathbf{r}-}$ are adapted. Taking now

$$\xi^+_\mathbf{r} = \frac{2 \cdot d^\lambda_{\mathbf{r}_-}}{(d^\lambda_\mathbf{r} + d^\lambda_{\mathbf{r}_-})^2} \text{ and } \xi^-_\mathbf{r} = \frac{2 \cdot d^\lambda_\mathbf{r}}{(d^\lambda_\mathbf{r} + d^\lambda_{\mathbf{r}_-})^2} \tag{5.9}$$

we get the update rules

$$\triangle \mathbf{w}_\mathbf{r} = \epsilon^+ \cdot \xi^+_\mathbf{r} \cdot \frac{f'|_{\mu_\lambda(\mathbf{r},\mathbf{v})} \cdot h_\gamma(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i})}{C(\gamma, K_{c_i})} \cdot \frac{\partial d^\lambda_\mathbf{r}}{\partial \mathbf{w}_\mathbf{r}} \tag{5.10}$$

$$\triangle \mathbf{w}_{\mathbf{r}_-} = -\epsilon^- \cdot \sum_{\mathbf{r}|\mathbf{w}_\mathbf{r} \in \mathbf{W}_c} \xi^-_\mathbf{r} \cdot \frac{f'|_{\mu_\lambda(\mathbf{r},\mathbf{v})} \cdot h_\gamma(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i})}{C(\gamma, K_{c_i})} \cdot \frac{\partial d^\lambda_{\mathbf{r}_-}}{\partial \mathbf{w}_{\mathbf{r}-}} \tag{5.11}$$

We remark that only for correct prototypes the neighborhood cooperativeness is applied.

Non-Standard Metrics, Relevance Learning and Metric Adaptation

We now consider relevance learning in the above introduced SNG, i.e. we study the influence of the parameter vector $\lambda = (\lambda_1, \ldots, \lambda_{D_V})$ in the distance measure $d^{\lambda}(\mathbf{v}, \mathbf{w})$ defined in (3.1). With other words, we are interested in adaptation of the distance measure in dependence on the λ_k to minimize the cost function. We are looking for the *relevance* of the parameters [15]. Then, an adaptation step for the parameters λ_k has to be added to the usual weight vector adaptation. Thereby, we assume that $\lambda_k \geq 0$ and $\sum_k \lambda_k = 1$.

For relevance learning of SRNG we get

$$\triangle\lambda_k = -2\epsilon_\lambda \sum_{\mathbf{r}|\mathbf{w}_\mathbf{r}\in\mathbf{W}_c} \frac{f'|_{\mu_\lambda(\mathbf{r},\mathbf{v})} \cdot h_\gamma(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i})}{C(\gamma, K_{c_i})} \cdot \frac{\partial\mu_\lambda(\mathbf{v})}{\partial\lambda_k} \tag{5.12}$$

using

$$\frac{\partial\mu_\lambda(\mathbf{v})}{\partial\lambda_k} = \xi^+_\mathbf{r}\frac{\partial d^\lambda_{\mathbf{r}_+}}{\partial\lambda_k} - \xi^-_\mathbf{r}\frac{\partial d^\lambda_{\mathbf{r}_-}}{\partial\lambda_k}. \tag{5.13}$$

followed by a renormalization. $\epsilon_\lambda > 0$ is the learning rate. For vanishing neighborhood cooperativeness $\gamma \to 0$ the SRNG turns to *Generalized Relevance LVQ (GRLVQ)* as relevance learning in GLVQ [62] with

$$\triangle\lambda_k = -\epsilon_\lambda \cdot f'|_{\mu_\lambda(\mathbf{r},\mathbf{v})} \cdot \frac{\partial\mu_\lambda(\mathbf{r},\mathbf{v})}{\partial\lambda_k} \tag{5.14}$$

Again as above, the update rule is also valid for the continuous case [123].

The learning rate ϵ should be in both approaches at least a magnitude smaller than the learning rates for the weight vectors. Then the weight vector adaptation takes place in a quasi stationary environment with respect to the (slowly) changing metric. Hence, the margin optimization takes place for each level of the parameter λ_k and we get an overall optimization of margin including a relevance weighting of the parameters.

As pointed out before, the similarity measure $d^{\lambda}(\mathbf{v}, \mathbf{w})$ is only required to be differentiable with respect to λ and $\mathbf{w}$. The triangle inequality has not to be fulfilled necessarily. This leads to a great freedom in the choice of suitable measures and allows the usage of non-standard metrics in a natural way. In particular, kernel based similarity measures are allowed [60]. In this way SNG/SRNG are comparable to SVMs [59].

In case that

$$d^{\lambda}(\mathbf{v}, \mathbf{w}) = \sum_{i=1}^{D_V} \lambda_i \cdot (v_i - w_i)^2 \tag{5.15}$$

is the squared, *scaled* Euclidean distance, whereby $\lambda_i \geq 0$ and $\sum_i \lambda_i = 1$, we immediately get in (5.5) and (5.6) for GRLVQ update:

$$\triangle\mathbf{w}_{\mathbf{r}_+} = \epsilon^+ \cdot 2 \cdot f'|_{\mu_\lambda(\mathbf{v})} \cdot \xi^+ \cdot \mathbf{\Lambda}\cdot(\mathbf{v} - \mathbf{w}_{\mathbf{r}_+}) \tag{5.16}$$

$$\triangle\mathbf{w}_{\mathbf{r}_-} = -\epsilon^- \cdot 2 \cdot f'|_{\mu_\lambda(\mathbf{v})} \cdot \xi^- \cdot \mathbf{\Lambda}\cdot(\mathbf{v} - \mathbf{w}_{\mathbf{r}_-}), \tag{5.17}$$

respectively, with $\mathbf{\Lambda}$ being the diagonal matrix with entries $\lambda_1, \ldots, \lambda_{D_V}$ and the relevance parameter update (5.14) in GRLVQ reads as

$$\triangle\lambda_k = -\epsilon_\lambda \cdot f'|_{\mu_\lambda(\mathbf{v})} \cdot \left(\xi^+(\mathbf{v}-\mathbf{w}_{\mathbf{r}_+})_k^2 - \xi^-(\mathbf{v}-\mathbf{w}_{\mathbf{r}_-})_k^2\right) \tag{5.18}$$

with $k = 1 \ldots D_V$.

In case of SRNG we get for the updates in case of the scaled Euclidean distance

$$\triangle\mathbf{w}_{\mathbf{r}_+} = 2\epsilon^+ \cdot \xi_{\mathbf{r}}^+ \cdot \frac{f'|_{\mu_\lambda(\mathbf{r},\mathbf{v})} \cdot h_\gamma\left(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i}\right)}{C\left(\gamma, K_{c_i}\right)} \cdot \mathbf{\Lambda}\cdot(\mathbf{v}-\mathbf{w}_{\mathbf{r}}) \tag{5.19}$$

$$\triangle\mathbf{w}_{\mathbf{r}_-} = -2\epsilon^- \cdot \sum_{\mathbf{r}|\mathbf{w}_{\mathbf{r}}\in\mathbf{W}_c} \xi_{\mathbf{r}}^- \cdot \frac{f'|_{\mu_\lambda(\mathbf{r},\mathbf{v})} \cdot h_\gamma\left(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i}\right)}{C\left(\gamma, K_{c_i}\right)} \cdot \mathbf{\Lambda}\cdot(\mathbf{v}-\mathbf{w}_{\mathbf{r}_-}), \tag{5.20}$$

with $\mathbf{\Lambda}$ being again the diagonal matrix with entries $\lambda_1, \ldots, \lambda_{D_V}$ and

$$\triangle\lambda_k = -\epsilon_\lambda \sum_{\mathbf{r}|\mathbf{w}_{\mathbf{r}}\in\mathbf{W}_c} \frac{f'|_{\mu_\lambda(\mathbf{r},\mathbf{v})} \cdot h_\gamma\left(\mathbf{r}, \mathbf{v}_i, \mathbf{W}_{c_i}\right)}{C\left(\gamma, K_{c_i}\right)} \cdot \left(\xi_{\mathbf{r}}^+ \cdot (\mathbf{v}-\mathbf{w}_{\mathbf{r}})_k^2 - \xi_{\mathbf{r}}^- \cdot (\mathbf{v}-\mathbf{w}_{\mathbf{r}_-})_k^2\right) \tag{5.21}$$

The derivations above refer to a global parametrized metric, optimized by a parameter vector λ. The usage of such a global parameterized metric is similar as within LDA, were a covariance matrix is estimated to optimize the global Mahalanobis metric involved in LDA. An improvement of LDA is given by QDA where the covariances are estimated locally for each single class. Taking this idea into account a local metric adaptation for GLVQ networks can be analyzed too.

5.2 SRNG and GRLVQ with Localized Metric Adaptation

Here we focus on *relevance* learning in LVQ as introduced in [62] referred as GRLVQ improved by local metric adaptation, similar as QDA, allows for local metric adaptation. GRLVQ is a local classifier scheme with a global scaled Euclidean metric. The former introduced notations are kept, but the changed derivations are given in the following. Again the task of vector quantization is implemented by the map Ψ as a winner-take-all rule thereby Ψ is now given locally as:

$$\Psi_{V\to\mathcal{A}}^{\lambda^{\mathbf{r}}} : \mathbf{v} \mapsto \mathbf{s}\left(\mathbf{v}\right) = \mathbf{argmin}_{\mathbf{r}\in A} d^{\lambda^{\mathbf{r}}}\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}}\right) \tag{5.22}$$

with $d^{\lambda^{\mathbf{r}}}\left(\mathbf{v}, \mathbf{w}\right)$ being an arbitrary differentiable distance measure depending on the parameter vector $\lambda^{\mathbf{r}}$ where r indicates the corresponding prototype vector. The neuron $\mathbf{s}$ is the winner or best matching unit, again.

LGRLVQ minimizes the cost function

$$Cost_{LGRLVQ} = \sum_{\mathbf{v}} f(\mu_{\mathbf{\Lambda}}(\mathbf{v})) \text{ with } \mu_{\mathbf{\Lambda}}(\mathbf{v}) = \frac{d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}_+} - d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-}}{d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}_+} + d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-}} \tag{5.23}$$

via stochastic gradient descent. Again $d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}_+}$ is the squared distance of the input vector $\mathbf{v}$ to the nearest prototype labeled with $c_{\mathbf{r}_+} = c_{\mathbf{v}}$, say $\mathbf{w}_{\mathbf{r}_+}$, and $d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-}$ is the squared distance to the best matching prototype but labeled with $c_{\mathbf{r}_-} \neq c_{\mathbf{r}_+}$, say $\mathbf{w}_{\mathbf{r}_-}$, localized with metric parameters $\lambda^{\mathbf{r}}$. In complete analogy, the learning rule of LGRLVQ is obtained by taking the derivatives. Using $\frac{\partial \mu_{\mathbf{\Lambda}}(\mathbf{v})}{\partial \mathbf{w}_{\mathbf{r}+}} = \xi_+ \frac{\partial d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}_+}}{\partial \mathbf{w}_{\mathbf{r}+}}$ and $\frac{\partial \mu_{\mathbf{\Lambda}}(\mathbf{v})}{\partial \mathbf{w}_{\mathbf{r}-}} = \xi_- \frac{\partial d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-}}{\partial \mathbf{w}_{\mathbf{r}-}}$ with

$$\xi^+ = \frac{2 \cdot d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-}}{(d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}_+} + d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-})^2},\ \xi^- = \frac{2 \cdot d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}+}}{(d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}_+} + d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-})^2} \tag{5.24}$$

one obtains for the weight updates:

$$\triangle \mathbf{w}_{\mathbf{r}_+} = \epsilon^+ \cdot f'|_{\mu_{\mathbf{\Lambda}}(\mathbf{v})} \cdot \xi^+ \cdot \frac{\partial d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}_+}}{\partial \mathbf{w}_{\mathbf{r}+}},\ \triangle \mathbf{w}_{\mathbf{r}_-} = -\epsilon^- \cdot f'|_{\mu_{\mathbf{\Lambda}}(\mathbf{v})} \cdot \xi^- \cdot \frac{\partial d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-}}{\partial \mathbf{w}_{\mathbf{r}-}} \tag{5.25}$$

For the adaptation of the parameters λ^r_k we get relevance learning

$$\triangle \lambda^r_i = -\epsilon \cdot f'|_{\mu_{\mathbf{\Lambda}}(\mathbf{v})} \cdot \frac{\partial \mu_{\mathbf{\Lambda}}(\mathbf{v})}{\partial \lambda^r_i},\ \frac{\partial \mu_{\mathbf{\Lambda}}(\mathbf{v})}{\partial \lambda^r_i} = \xi^+ \frac{\partial d^{\lambda^{\mathbf{r}_+}}_{\mathbf{r}_+}}{\partial \lambda^r_i} - \xi^- \frac{\partial d^{\lambda^{\mathbf{r}_-}}_{\mathbf{r}_-}}{\partial \lambda^r_i}.$$

$\epsilon^+, \epsilon^-, \epsilon > 0$ are learning rates.

In case of $d^{\lambda^{\mathbf{k}}}_{\mathbf{k}} = \sum_{i=1}^{D_V} \lambda^k_i \cdot (v_i - w_{ki})^2$ is the squared, *scaled* Euclidean distance, whereby $\lambda^k_i \geq 0$ and $\sum_i \lambda^k_i = 1$, we have in (5.25)

$$\triangle \mathbf{w}_{\mathbf{r}_+} = \epsilon^+ \cdot 2 \cdot f'|_{\mu_{\mathbf{\Lambda}}(\mathbf{v})} \cdot \xi^+ \cdot \lambda^{\mathbf{r}_+} \cdot (\mathbf{v} - \mathbf{w}_{\mathbf{r}_+}) \tag{5.26}$$

$$\triangle \mathbf{w}_{\mathbf{r}_-} = -\epsilon^- \cdot 2 \cdot f'|_{\mu_{\mathbf{\Lambda}}(\mathbf{v})} \cdot \xi^- \cdot \lambda^{\mathbf{r}_-} \cdot (\mathbf{v} - \mathbf{w}_{\mathbf{r}_-}), \tag{5.27}$$

respectively, with $\lambda^{\mathbf{r}}$ being the diagonal matrix with entries $\lambda^r_1, \ldots, \lambda^r_{D_V}$ and (5.14) reads as

$$\triangle \lambda^r_i = -\epsilon \cdot f'|_{\mu_{\mathbf{\Lambda}}(\mathbf{v})} \cdot \left(\xi^+ (\mathbf{v} - \mathbf{w}_{\mathbf{r}_+})^2_i - \xi^- (\mathbf{v} - \mathbf{w}_{\mathbf{r}-})^2_i\right) \tag{5.28}$$

with $i = 1 \ldots D_V$, followed by normalization. It should be emphasized again that LGRLVQ can be used with an individual metric $\lambda^{\mathbf{r}}$ for each prototype $\mathbf{w}_{\mathbf{r}}$ or with a class wise metric shared within prototypes with the same class label c_r. Nicely for one metric shared by all prototypes LGRLVQ gives GRLVQ. For local metric adaptation SRNG is referred as LSRNG and the derivations can be obtained in complete analogy as for LGRLVQ.

5.3 Different examples of problem specific metric adaptation

An early approach to data-specific distance calculations has been given by Gustafson and Kessel, who have extended the unsupervised fuzzy k-means algorithm by covariance matrices approximating the ellipsoidal radii of the k data clusters [52]. The computationally costly operations per iteration involve k local Mahalanobis distance and determinant calculations. The step away from the local distances to global metrics with adaptive parameters has been realized in several works, some of which are batch CTM [26], DSLVQ [112], and RLVQ [15]. These three methods share the common property that they heuristically adapt factors which weight the influence of dimensions on an Euclidean distance term. Sinkkonen and Kaski formulate a mathematically more rigorous approach by means of a function for the classification costs. By definition, this function takes its minimum for the best correspondence of the original data distribution and the distribution modeled with data prototypes, both related to auxiliary data labels [76]. Parameter adaptation can be based on information theoretic criteria, as done in Sinkkonen's and Kaski's stochastic expectation maximization framework [75].

Alternatively, the parameter update dynamic can be driven by distortion criteria reflected by the cost function for the data and their prototypes. If this function is differentiable, optimum parameters for its minimization can be iteratively obtained with tools from calculus. Adaptive factors for weighting individual data attributes have been first introduced for a cost function variant of LVQ by Hammer and Villmann [62].

5.3.1 Scaled Euclidean Metric in LVQ

Recently, Hammer et al. have linked generalized relevance LVQ (GRLVQ) to the structural risk minimization framework of learning theory [58]. Following ideas of Crammer et al. [33], it has been shown that the GRLVQ algorithm with adaptive diagonal Euclidean metric restricted to $\sum_{i=1}^{d} |\lambda_i| = 1$ leads to margin maximization. Apart from the Euclidean distance, more generally, any kernelized version leads to margin maximization. This property makes GRLVQ competitive to the well-established support vector machine (SVM) classifier [29, 134]. Since SVM usually generates larger and computationally more demanding models of the data than GRLVQ [57], this promising LVQ variant will be further studied in the following, and extensions to more general metrics and to initialization-tolerant prototype cooperation will be considered.

The simplest metric structure to be plugged into the update equations is a generalized weighted Euclidean distance:

$$d_{\lambda}^{\mathrm{EUC}}(\mathbf{v}, \mathbf{w_r}) = \sum_{i=1}^{d} \lambda_i^{b_\lambda} (v_i - w_r^i)^{b_w}, \quad \text{integers } b_\lambda, b_w \geq 0, b_w \text{ even}$$

The exponent b_w controls how much the mismatch in single dimensions contributes to

the cost: large exponents lead to the emphasis of outlier dimensions, whereas small values better tolerate deviations that might be subject to noise. In simple words: large b_w focus on dimensions with large differences and, compared to this, small b_w focus on dimensions with small differences. In the squared case with $b_w = 2$, the derivative for the prototype update $2 \cdot (v_i - w_r^i)$ is recognizable as Hebbian learning term. The exponent b_λ takes influence on the adaptation dynamic of the factors λ_i; typical values for b_λ are $0, 1, 2,$ and 4. Disregarding the metric parameters $\lambda_i > 0$ by setting $b_\lambda = 0$ yields the original GLVQ. If the metric parameters λ_i are taken into account for $b_\lambda \neq 0$, they become factors that weight the influence of differences in certain data dimensions. For example, a noise dimension added to a classification task usually does not help to drive the prototypes into the clusters where they belong: since the position of a Voronoi decision boundary between two classes depends on every dimension of the involved two prototypes, noisy components induce an orientation of the borderline different from the one for omitted noise dimensions; this noise-affected borderline may lead to the accidental inclusion of data points with wrong labels.

Throughout this work, metrics are used for which their variables λ_i refer to the weighting of dimension i, although a more general parameter utilization is also possible. The forced normalization to $\sum_{i=1}^{d} |\lambda_i| = 1$ is necessary, because the dynamic can drive diverging relevance factors to extreme values. For an odd exponent, such as $b_\lambda = 1$, an additional clipping of the λ_i to non-negative values must implemented, because the corresponding derivative of constant 1, scaled by the metric learning rate, can decrease the λ_i to undesired negative values. This clipped normalization has been performed during each step of the training update; thus, a comparison of the different relevance profiles is possible after training. Putting together normalized adaptive metric parameters for $b_\lambda = 1$ and GLVQ yields generalized relevance LVQ, GRLVQ for short [62].

As pointed out in [105] the choice of the exponents b_w and b_λ remains to be discussed. Usually, the quadratic distortion measure d^2 with $b_w = 2, b_\lambda = 0$ is applied which yields invariance to data rotations and which provides risk minimization for false classification of Gaussian data. For data attributes with incompatible semantics, the rotation invariance might be unwanted; instead, the intra-dimensional matching should outperform the inter-dimensional fitting. Additionally, it is known that Gaussian data can be described by the first two central moments, the mean and the variance; for non-Gaussian data, though, moments of higher order, expressed by larger metric exponents, are more appropriate for capturing data statistics such as sharp cluster boundaries [100, 9]. As a matter of fact, in many experiments the quartic distance d^4 with $b_w = 4, b_\lambda = 2$ has turned out to be a reliable metric, leading faster to classification results better than the d^2 above and also better than its adaptive counterpart $d^2_{\lambda^2}$ with $b_w = 2, b_\lambda = 2$. Its derivative with respect to the prototype locations displays a Hebbian term, taken to the power of three; thus, prototypes with badly matching dimensions are nonlinearly dragged towards the centroids of the according attribute, which yields an alignment along the data axes, in contrast to the rotation invariant standard Euclidean metric. Empirically, a choice of larger exponents b_w and b_λ has

not further improved the results, but training tends to get numerically unstable. Recent experiments show that using $b_\lambda = 1$ with clipping $\lambda_i \geq 0$ leads to better results than $b_\lambda > 1$ with built-in self-inhibition: higher exponents tend to boost the most relevant dimensions at the expense of less important, yet contributing attributes. By means of adaptive metric parameters adjusted to minimize the cost function, the importance of data dimensions for the classification task are determined. After the training λ reflects a dimension relevance profile. This can be used to identify sufficiently large λ_i indicating attributes onto which the data set might be projected for further classification, analysis, or for visualization. Apart from this dimensionality reduction, another important benefit of the relevance learning is the generalization improvement of the networks: since noise dimensions are rated less important in the training data, false influences will be canceled also during the classification of new data. This way, a direct connection is established between network pruning and relevance learning.

5.3.2 Mahalanobis Metric in LVQ

Related to the weighted Euclidean distance, the more general Mahalanobis metric can be computed. In case of uncorrelated dimensions, we end up in the GRLVQ metric with the squared Euclidean metric. The (parametrized) Mahalanobis distance is given by

$$d^\lambda(\mathbf{v},\mathbf{w}) = (\mathbf{v}-\mathbf{w})\,\mathbf{C}_\lambda^{-1}\,(\mathbf{v}-\mathbf{w})^T = \mathbf{m}_\mathbf{r}^\lambda \tag{5.29}$$

with $\mathbf{C}_\lambda$ being the parametrized covariance matrix defined as follows: Let $\mathbf{C} = \check{\mathbf{P}}\check{\mathbf{D}}\check{\mathbf{P}}^T$ be the diagonal representation of the usual covariance matrix with diagonal matrix $\check{\mathbf{D}}$. Then the parametrized diagonal matrix $\check{\mathbf{D}}_\lambda$ is obtained as $\check{\mathbf{D}}_\lambda = \mathbf{\Lambda}\cdot\check{\mathbf{D}}$ with $\mathbf{\Lambda}$ is the diagonal matrix and the diagonal elements $\Lambda_{ii} = \lambda_i$ generate the parameter vector λ. Finally we define analogously[2]:

$$\mathbf{C}_\lambda^{-1} = \mathbf{P}\mathbf{D}_\lambda\mathbf{P}^T \tag{5.30}$$

Using the decomposition of $\mathbf{C}_\lambda^{-1}$ we can rewrite the distance as follows:

$$\begin{aligned} d^\lambda(\mathbf{v},\mathbf{w}) &= ((\mathbf{v}-\mathbf{w})\cdot\mathbf{P})\mathbf{D}_\lambda(\mathbf{P}^T\cdot(\mathbf{v}-\mathbf{w})^T) && (5.31)\\ &= \iota\cdot(\mathbf{D}_\lambda((\mathbf{v}-\mathbf{w})\cdot\mathbf{P})\cdot(\mathbf{P}^T\cdot(\mathbf{v}-\mathbf{w})^T)^T)^T && (5.32)\\ &= \iota\cdot(\mathbf{D}_\lambda((\mathbf{v}-\mathbf{w})\cdot\mathbf{P})^2)^T && (5.33)\\ \text{with } \iota &= (1_1,\ldots,1_{D_V}) && (5.34) \end{aligned}$$

We immediately get in (5.5) and (5.6) for GLVQ update in Mahalanobis space:

$$\triangle\mathbf{w}_{\mathbf{r}_+}^m = \epsilon^+\cdot 2\cdot f'|_{\mu_\lambda(\mathbf{v})}\cdot\xi^+\cdot\frac{\partial\mathbf{m}_{\mathbf{r}_+}^\lambda}{\partial\mathbf{w}_+} \tag{5.35}$$

[2] it should be noted that $\mathbf{P}^T = \mathbf{P}^{-1}$

$$\triangle \mathbf{w}_{\mathbf{r}_-}^m = -\epsilon^- \cdot 2 \cdot f'|_{\mu_\lambda(\mathbf{v})} \cdot \xi^- \cdot \frac{\partial \mathbf{m}_{\mathbf{r}_-}^\lambda}{\partial \mathbf{w}_-} \tag{5.36}$$

respectively, with $\frac{\partial \mathbf{m}_{\mathbf{r}_+}^\lambda}{\partial \mathbf{w}_+}$ defined as:

$$\frac{\partial \mathbf{m}_{\mathbf{r}}^\lambda}{\partial \mathbf{w}_+} = -2 \cdot \mathbf{D}_\lambda \cdot ((\mathbf{v} - \mathbf{w}_+) \cdot \mathbf{P})^T \tag{5.37}$$

$$\frac{\partial \mathbf{m}_{\mathbf{r}}^\lambda}{\partial \mathbf{w}_-} = -2 \cdot \mathbf{D}_\lambda \cdot ((\mathbf{v} - \mathbf{w}_-) \cdot \mathbf{P})^T \tag{5.38}$$

the update is applied in the Mahalanobis space on the transformed prototypes followed by an inverse transformation using $(\mathbf{D}_\lambda \cdot \mathbf{P}^T)^{-1}$. The update of the relevance factors is derived as:

$$\frac{\partial \mathbf{m}_{\mathbf{r}}^\lambda}{\partial \lambda} = -\epsilon_\lambda \cdot f'|_{\mu_\lambda(\mathbf{r},\mathbf{v})} \cdot \left(\xi_{\mathbf{r}}^+ \cdot \frac{\partial d_{\mathbf{r}}^\lambda}{\partial \lambda_k} - \xi_{\mathbf{r}}^- \cdot \frac{\partial d_{\mathbf{r}_-}^\lambda}{\partial \lambda_k} \right) \tag{5.39}$$

with

$$\tag{5.40}$$

$$\frac{\partial d_{\mathbf{r}+}^\lambda}{\partial \lambda_k} = \mathbf{D} \cdot (((\mathbf{v} - \mathbf{w}_+)^T \cdot P)^2)^T \tag{5.41}$$

$$\frac{\partial d_{\mathbf{r}-}^\lambda}{\partial \lambda_k} = \mathbf{D} \cdot (((\mathbf{v} - \mathbf{w}_-)^T \cdot P)^2)^T \tag{5.42}$$

The application of this measure allows a relevance ranking of the principal components of the data distribution for the given classification task. The Mahalanobis distance depends on the inverse of the covariance matrix of the underlying data set and with respect to the dataset the determination of the inverse may be numerically ill-conditioned which may lead to instable behavior. In this case one may substitute the diagonal matrix by a pseudo inverse. Recently, an application of such a distance measure to the k-means clustering method has been given by Xing et al. [181]. In the present work, the Mahalanobis metric has been integrated to GRLVQ and applied successfully in the classification of overlapping data [172].

5.3.3 Tanimoto Metric in LVQ

We now introduce the Tanimoto distance measure ([40]) as an alternative correlative measure in LVQ networks. This distance measure is well known from taxonomic research and has been applied successfully in structural domains such as distance determination on strings and compound detection in chemical research. It has been initially introduced by Tanimoto ([148, 118], a nice overview is given in [107]). The basic

idea is as follows; consider the number of equal pairs of corresponding attributes of two objects. If these are the complete set of attributes of the two objects, the objects would be identical. The presence of one additional unequal pair of corresponding attributes would destroy the identity relationship, but would leave the objects quite similar. More and more additional pairs of unequal attributes would increase the distance between these objects. Tanimoto proposed that the ratio of the number of pairs of non-zero identical corresponding attributes to the total number of pairs of corresponding attributes, of which at least one member is not zero, be defined as the Similarity Coefficient. This yields a value of 1 for identical objects and 0 for extremely dissimilar objects and is similar to a probability (in this as well as other respects) [107].

This similarity coefficient was originally defined by Tanimoto for attributes which have magnitudes of either 0 or 1 (the binary case). He has extended it to cover the continuous case. The Tanimoto distance measure for continues variables is defined as:

$$t(\mathbf{v},\mathbf{w}) = \frac{d(\mathbf{v},\mathbf{w})}{d(\mathbf{v},\mathbf{v}) + d(\mathbf{w},\mathbf{w}) - d(\mathbf{v},\mathbf{w})} \tag{5.43}$$

With $d(\mathbf{v},\mathbf{w})$ as the inner product. For GLVQ networks we will use $1.0 - t(\mathbf{v},\mathbf{w})$ as the final distance measure and constrain the data to exist $\in [0,1]$ which is no strong limitation but keeps the original principle of the Tanimoto measure.

We immediately get in (5.5) and (5.6) for GLVQ update:

$$\triangle \mathbf{w}_{\mathbf{r}_+} = -\epsilon^+ \cdot 2 \cdot f'|_{\mu(\mathbf{v})} \cdot \xi^+ \cdot \frac{\partial \mathbf{t}_{\mathbf{r}_+}}{\partial \mathbf{w}_+} \tag{5.44}$$

$$\triangle \mathbf{w}_{\mathbf{r}_-} = \epsilon^- \cdot 2 \cdot f'|_{\mu(\mathbf{v})} \cdot \xi^- \cdot \frac{\partial \mathbf{t}_{\mathbf{r}_-}}{\partial \mathbf{w}_-} \tag{5.45}$$

respectively, with $\frac{\partial \mathbf{t}_{\mathbf{r}_+}}{\partial \mathbf{w}_+}$ and resp. $\frac{\partial \mathbf{t}_{\mathbf{r}_-}}{\partial \mathbf{w}_-}$ defined as:

$$\frac{\partial \mathbf{t}_{\mathbf{r}}}{\partial \mathbf{w}} = \frac{\mathbf{v}\overbrace{(d(\mathbf{v},\mathbf{v}) + d(\mathbf{w},\mathbf{w}) - d(\mathbf{v},\mathbf{w}))}^{\zeta}}{(d(\mathbf{v},\mathbf{v}) + d(\mathbf{w},\mathbf{w}) - d(\mathbf{v},\mathbf{w}))^2} - \frac{d(\mathbf{v},\mathbf{w})(2\cdot\mathbf{w} - \mathbf{v})}{(d(\mathbf{v},\mathbf{v}) + d(\mathbf{w},\mathbf{w}) - d(\mathbf{v},\mathbf{w}))^2} \tag{5.46}$$

$$= \frac{\zeta\mathbf{v} + (d(\mathbf{v},\mathbf{w})\,\mathbf{v} - 2d(\mathbf{v},\mathbf{w})\,\mathbf{w})}{\zeta^2} = \frac{[(d(\mathbf{v},\mathbf{w}) + \zeta)\mathbf{v} - 2d(\mathbf{v},\mathbf{w})\,\mathbf{w}]}{\zeta^2} \tag{5.47}$$

$$= \frac{1}{\varrho\zeta}[(1+\varrho)\mathbf{v} - 2\mathbf{w}]) \qquad \text{with } \varrho = \frac{\zeta}{d(\mathbf{v},\mathbf{w})} \tag{5.48}$$

with $\mathbf{w}$ having the correct or incorrect label.

5.3.4 Correlative Feature Elimination with LVQ networks

A well known problem in classification is the reduction of the dimensionality n of the feature space (in our case the peak areas) to overcome the risk of *overfitting*. Data

overfitting arises, when the number n of features is large with respect to the number l of training samples. In such a situation, one can easily find a decision function that separates the training data but will perform poorly on test data. For a review of feature selection, see e.g. [81, 73, 108]. Exhaustive enumeration is impractical for large numbers of features because of the combinatorial explosion of the number of subsets. Therefore, feature selection is often done using greedy methods. Among various possible methods feature-ranking techniques are particularly attractive. Thereby methods which only consider one single feature at time are often not sufficient because features that individually do not separate well but are useful in combination with other features are missed.

The current approaches of metric adaptation allow for modifying the used metric to be substituted by some L^2 metric, by e.g. a correlation measure [105] or as introduced above the Mahalanobis metric. If we consider some kind of scaled Euclidean metric we end up with a ranking of individual feature dimensions. Thereby the individual dimensions are considered mainly to be independent. This however may not be the general case and especially for MS data a dependence between different peaks with underlying fractions of the same protein may occur. To overcome this we introduce the so called correlative feature elimination which is similar to the method as introduced in [53]. Thereby we encapsulate the classification estimation by a feature selection method which makes use of the calculated feature ranking. A good feature ranking criterion however is not necessarily a good feature subset ranking criterion [53]. To overcome this in [53] a feature elimination is proposed which will be adapted in the following.

1. Initialization of e.g. SRNG
2. Let $M = \{1, \ldots, D_v\}$ be the whole set of feature indices
3. Let $R = \{\}$ a later feature ranking
4. Initialize the relevance vector with equal weights for each dimension with $\sum_i \lambda_i = 1$
5. REPEAT (as long $\#M > 0$)
   - Train SRNG network with the training data using feature indices i with $i \in M$ (the remaining indices maybe scaled by 0.0)
   - Let λ the feature ranking from the trained SRNG
   - Determine index for worst feature $f_w = \mathbf{argmin}\{\lambda\}$
   - Add feature to the ranking: R = $\{R, f_w\}$ and remove feature from $M = M/f_w$
6. The final ranking with incorporation of correlative dependences is stored in R

5.4 Experiments and Results

We show the capabilities of LGRLVQ on a synthetic set of 1200 data points in four classes. Each class consists of 300 data points in two dimensions and the data is Gaussian distributed with different variances per data class. The obtained dataspace is shown in Figure 5.1 (left). We use the LGRLVQ to learn a model for the prototype

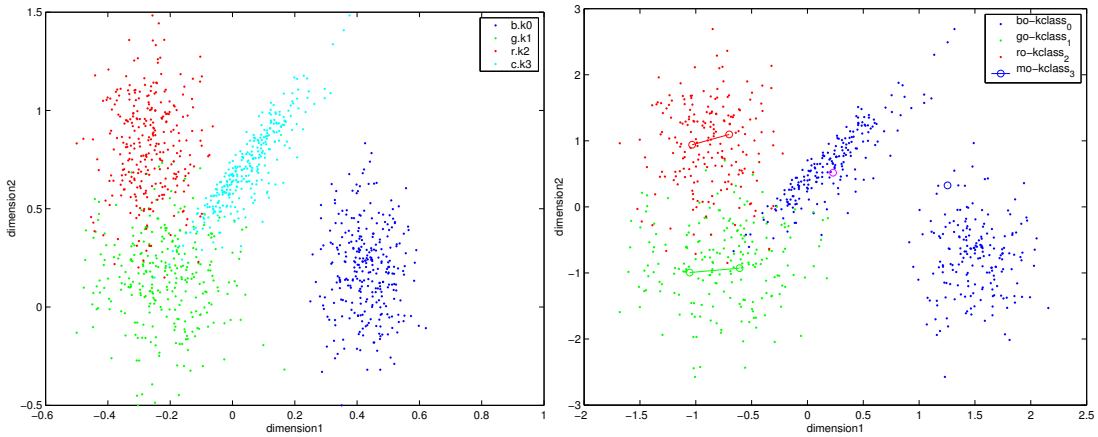


Figure 5.1: The left figure shows a plot of the artificial data sets. In the right figure a normalized pseudo plot (without lambda scaling) of the data sets and there relative codebook positions is given. Prototypes of the same class have been connected.

distribution with localized metric adaptation. We use 2 prototypes per class. Here we have relabeled the class with label c.k3 to b.k0 (c.f. Figure (5.1)) to show the subclass identification by local metric adaptation. The data were split randomly into a training and a test set for a simple crossvalidation procedure. The prototypes are randomly initialized and are then optimized in each step in the prototype positions and the codebook metric. It should be noted that all classes (without the one with label b.k0) are overlapping and therefore not linear separable. The LGRLVQ algorithm optimized the prototype distribution within 3000 steps and obtained finally 93% prediction upon the given test set. In Table 5.1 we show the obtained relevance profiles per prototype.

Considering the relevance profiles as well as the receptive fields it is clearly pointed out that the LGRLVQ algorithm has successfully learned the data distributions and the prototypes has been adapted well. For the class b.k0 we recognize two effects. On one side we see a clear highlighting of the first dimension (namely the axis x) this is true for both prototype with the label b.k0. On the other hand we also see that the obtained relevance profiles show a clear difference in the axis y. This second dimension is less important as clearly visible in Figure 5.1 but the prototype located in the cloud of class b.k0 has nearly pruned this dimension whereas the prototype responsible for the cloud with the original label c.k3 has still an importance of 7% which means it is seven hundred times more important than for b.k0. This can be seen as an indicator

Class label	Prototype	Relevance 1st dim	Relevance 2nd dim
b.k0 (c.k3)	1	0.93	0.07
b.k0	2	0.99	0.0001
r.k1	1	0.91	0.09
r.k1	2	0.91	0.09
g.k2	1	0.92	0.08
g.k2	2	0.90	0.10

Table 5.1: Relevance profiles obtained from LGRLVQ for data as shown in Figure 5.1

for a subclass which indeed is the case as synthetically established in the experimental setup and therewith was recovered successfully by LGRLVQ. For the other prototype pairs we recognize that they are quite similar and homogeneous.

In Figure 5.1 we show a pseudo plot (without lambda scaling) of the data sets with the obtained relative prototype positions. Here it is also obvious (ignoring some scaling effects) that the prototypes are located well in the data distribution they belongs to and that the artificial subclass of class b.k0 is represented by the prototype $b.k0_2$ which is in fact responsible for the class c.k3 as already recognized in Table 5.1.

Now some experiments for the formerly introduced metric adaptations are given, starting with a small example for the Tanimoto metric. As a simple example we may consider three patterns (v_a, v_b, v_c):

$$v_a = (1.00, 0.01, 0.01, 0.01) \tag{5.49}$$
$$v_b = (0.01, 0.01, 1.00, 0.01) \tag{5.50}$$
$$v_c = (1.00, 0.01, 1.00, 0.01) \tag{5.51}$$

The three patterns differ to each other, thereby we see that the number of differences $D(v_i, v_j)$ between the patterns is given as $D(v_a, v_b) = 2, D(v_a, v_c) = 1, D(v_b, v_c) = 1$. Now we generated an artificial data set of three classes, each with 100 data points. The first class takes items which are based upon v_a with added random noise and equivalent for class two and three based on v_b and v_c respectively. Using LVQ with the Tanimoto measure we obtained a prediction accuracy of 91% with respect to 85% for SNG with Euclidean metric using 1 prototype per class. The obtained codebook is shown in Table (5.2).

The Mahalanobis distance aims on a PCA similar behavior, were direction of highest variance get more importance than directions with low variance explanation. To analyse the Mahalanobis metric in LVQ networks the synthetic scatter data have been used. We applied the LVQ with Mahalanobis metric (LVQ-MD) on the data using 200 cycles and 16 prototypes. The initial two dimensional data grid is shown in Figure (5.2). After applying the LVQ-MD an inverse covariance metric and a vector with scaling values for the eigenvalues is obtained. After 200 cycles the metric was modified

Label	dim_1	dim_2	dim_3	dim_4
1	0.582	0.005	0.007	0.004
2	0.008	0.006	0.647	0.006
3	0.701	0.006	0.593	0.006

Table 5.2: Codebook for GLVQ with Tanimoto distance. We see the recognition of the basic patterns v_a, v_b, v_c

in a way that the original data, given in a rectangular form were scaled in a trapezoidal structure, so that the common classes are recognizable on a diagonal line with orientation to the axis of highest variance. This reflect the PCA behavior of the metric and shows the successful application (c.f. Figure (5.2)). The obtained final affine transformation matrix is given as:

$$\begin{pmatrix} 0.6192 & -0.3704 \\ -0.3704 & 0.3605 \end{pmatrix}$$

The obtained solution simplifies the original classification problem to assign the correct prototype to each data cluster for the given chess pattern structure to a problem were the different classes are given in one diagonal row with interchanging class labels. The obtained model was suitable to get a 100% prediction in a crossvalidation procedure.

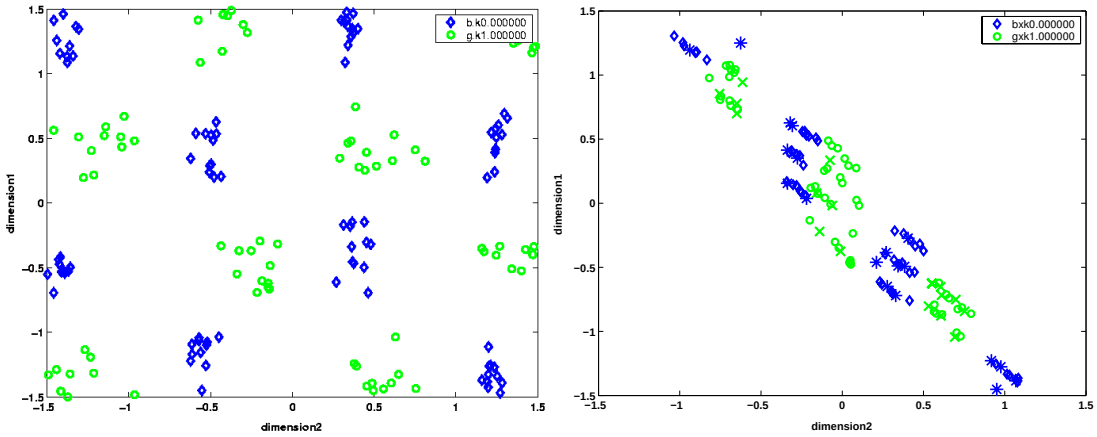


Figure 5.2: The left figure shows a plot of the artificial data sets. In the right figure the same plot with applied Mahalanobis metric is given. The two classes are given as circles and diamonds and the corresponding prototypes are given by 'x' and '*'.

We now consider the correlative feature elimination in comparison to the ordinary relevance profile and its subsequently derived naive ranking. The naive ranking orders features according to their individual relevance. The CFE is a feature subset

ranking. The nested feature subsets contain complementary features not necessarily individually most relevant. This is related to the relevance vs. usefulness distinction as noted in [81, 53]. This distinction is most important in the case of correlated features. Imagine, for example a two class classification problem with 7 features, but only 3 distinct features, whereby two of them are both equally useful and the third can be considered to be random to distinguish the different samples. Hence we may get a feature vector like this $(x_1, x_1, x_2, x_2, x_2, x_2, x_3)$. A naive ranking using SRNG resulted in $\{x_1(1/3), x_1(1/3), x_2(1 \cdot 10^{-3}), x_2(1 \cdot 10^{-3}), x_2(1 \cdot 10^{-3}), x_2(1 \cdot 10^{-3}), x_3(1/3 - 4 \cdot 10^{-3})\}$, assuming that the ranking criterion gives equal weight magnitudes to identical features. In the synthetic data $x_1, x_2 \in \{0, 1, \ldots, 6\}$ and $x_3 \in [0, 1]$ with $||x_1 - x_2|| = 2$ for class 1 and $||x_1 - x_2|| = 3$ for class 2. If we select a subset of two features in accordance to the naive ranking, we eliminate the useful feature x_2 and incur possible classification performance degradation. In contrast a CFE run using SRNG will produce (per iteration):

1. $\{x_1(1/3), x_1(1/3), x_2(10^{-3}), x_2(10^{-3}), x_2(10^{-3}), x_2(10^{-3}), x_3(1/3 - 4 \cdot 10^{-3})\}$
2. $\{x_1(32/100), x_1(32/100), x_2(0), x_2(0), x_2(0), x_3(36/100)\}$
3. $\{x_1(44/100), x_1(44/100), x_2(0), x_2(0), x_3(12/100)\}$
4. $\{x_1(1/100), x_1(1/100), x_2(6/10 - 2/100), x_3(4/10)\}$
5. $\{x_1(36/100), x_2(63/100), x_3(1/100)\}$
6. $\{x_1(4/10), x_2(6/10)\}$
7. $\{x_1(1)\}$

Therefore if we select two features according to CFE, we obtain both x_1 and x_2, as desired, however it should be mentioned that the CFE ranking is not unique if different equally ranked features are available.

Chapter 6

Supervised Fuzzified Learning Vector Quantization

You're one hundred percent positive that the ship which is crashed on the bottom of this ocean is the ship which you said you were one hundred percent positive could one hundred percent positively never crash?
Douglas Adams

The supervised methods considered so far determine the classification decision only as a crisp label. Especially for classes with overlapping data distributions the classification decision should not just be crisp but also give insight in the safety of the classification decision. This is especially important in case of clinical data were some data items maybe located close to the decision boundary, and then the application of further tests maybe be adequate. Here it would be desirable to get some additional information of the classifier, indicating this situation. Recently a new prototype based classifier [135] has been proposed which can be improved by fuzziness. First the basic notations are given.

6.1 Self Adapted Soft Nearest Prototype Classification

Soft Nearest Prototype Classification (SNPC) has been proposed as alternative to the standard LVQ 2.1 algorithm based on another cost function. It introduces soft assignments for data vectors to the prototypes in order to obtain a cost function for classification such that adaptation forms a gradient descent. In the standard variant of SNPC provided in [135] one considers

$$E\left(\mathcal{S},\mathcal{W}\right)=\frac{1}{N_{\mathcal{S}}}\sum_{k=1}^{N_{\mathcal{S}}}\sum_{\mathbf{r}}u_{\tau}\left(\mathbf{r}|\mathbf{v}_{k}\right)\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_{k}}}\right) \tag{6.1}$$

as the cost function with $\mathcal{S}=\{(\mathbf{v},c_{\mathbf{v}})\}$ the set of all input pairs. $N_{\mathcal{S}}=\#\mathcal{S}$, $\mathbf{W}=\{\mathbf{w}_{\mathbf{r}}\}$ is the set of all codebook vectors and $\mathcal{W}=\{(\mathbf{w}_{\mathbf{r}},c_{\mathbf{r}})\}$ whereby $c_{\mathbf{r}}$ is the class label of

$\mathbf{w_r}$, as before. The value $\alpha_{\mathbf{r},c_{\mathbf{v}_k}}$ equals the unit if $c_{\mathbf{v}_k} = c_{\mathbf{r}}$. $u_\tau(\mathbf{r}|\mathbf{v}_k)$ is the probability that the input vector $\mathbf{v}_k$ is assigned to the prototype $\mathbf{r}$. In case of the *winner-takes-all* mapping (3.1) one obtains $u_\tau(\mathbf{r}|\mathbf{v}_k) = \delta(\mathbf{r} = \mathbf{s}(\mathbf{v}_k))$ the usual crisp LVQ.

In order to minimize the cost function (6.1) in [135] the variables $u_\tau(\mathbf{r}|\mathbf{v}_k)$ are taken as soft assignments. This allows a gradient descent. As proposed in [135], the probabilities (soft assignments) are chosen to be of normalized exponential form

$$u_\tau(\mathbf{r}|\mathbf{v}_k) = \frac{\exp\left(-\frac{d(\mathbf{v}_k,\mathbf{w_r})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\exp\left(-\frac{d(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)} \tag{6.2}$$

whereby d is the distance measure used in (3.1). Then the cost function (6.1) can be rewritten into

$$E_{soft}(\mathcal{S},\mathcal{W}) = \frac{1}{N_\mathcal{S}}\sum_{k=1}^{N_\mathcal{S}} lc\left((\mathbf{v}_k, c_{\mathbf{v}_k}), \mathbf{W}\right) \tag{6.3}$$

with local costs

$$lc\left((\mathbf{v}_k, c_{\mathbf{v}_k}), \mathbf{W}\right) = \sum_{\mathbf{r}} u_\tau(\mathbf{r}|\mathbf{v}_k)\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) \tag{6.4}$$

i.e., the local error is the sum of the class assignment probabilities $\alpha_{\mathbf{r},c_{\mathbf{v}_k}}$ to all prototypes of an incorrect class. Hence,

$$lc\left((\mathbf{v}_k, c_{\mathbf{v}_k}), \mathbf{W}\right) \leq 1 \tag{6.5}$$

Because the local costs $lc\left((\mathbf{v}_k, c_{\mathbf{v}_k}), \mathbf{W}\right)$ are continuous and bounded, the cost function (6.3) can be minimized by stochastic gradient descent using the derivative only of the local costs:

$$\triangle\mathbf{w_r} = \begin{cases} \frac{1}{2\tau^2}u_\tau(\mathbf{r}|\mathbf{v}_k)\cdot lc\left((\mathbf{v}_k, c_{\mathbf{v}_k}), \mathbf{W}\right)\cdot\frac{\partial d_\mathbf{r}}{\partial \mathbf{w_r}} & \text{if} \quad c_{\mathbf{v}_k} = c_\mathbf{r} \\ -\frac{1}{2\tau^2}u_\tau(\mathbf{r}|\mathbf{v}_k)\cdot\left(1 - lc\left((\mathbf{v}_k, c_{\mathbf{v}_k}), \mathbf{W}\right)\right)\cdot\frac{\partial d_\mathbf{r}}{\partial \mathbf{w_r}} & \text{if} \quad c_{\mathbf{v}_k} \neq c_\mathbf{r} \end{cases} \tag{6.6}$$

using

$$\frac{\partial lc}{\partial \mathbf{w_r}} = -u_\tau(\mathbf{r}|\mathbf{v}_k)\left(\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) - lc\left((\mathbf{v}_k, c_{\mathbf{v}_k}), \mathbf{W}\right)\right)\cdot\frac{\partial d_\mathbf{r}}{\partial \mathbf{w_r}} \tag{6.7}$$

This leads to the learning rule

$$\mathbf{w_r} = \mathbf{w_r} - \epsilon(t)\cdot\triangle\mathbf{w_r} \tag{6.8}$$

with learning rate $\epsilon(t)$ fulfilling the relations $\sum_{t=0}^{\infty}\epsilon(t) = \infty$ and $\sum_{t=0}^{\infty}(\epsilon(t))^2 < \infty$ as usual.

A window rule like for standard LVQ2.1 can be derived for SNPC, too. The window rule is necessary for numerical stabilization as for standard LVQ2.1 [83],[135]

and improves the learning speed. One has to update all weights for which the local value

$$\eta_{\mathbf{r}} = lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right) \cdot \left(1 - lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)\right) \tag{6.9}$$

is less than the threshold value η with

$$0 \ll \eta < 0.25 \tag{6.10}$$

which is similar to the window rule in LVQ2.1.

6.2 Fuzzified Self Adapted Soft Nearest Prototype Classification

For both SNPC and GLVQ crisp prototype labels have to be defined and kept fix during adaptation. Now we will relax this restriction. In *Dynamic Fuzzy Labeling* for SNPC (FSNPC) we allow dynamic fuzzy labels $\alpha_{\mathbf{r},c}$ to indicate the responsibility of weight vector $\mathbf{w_r}$ to class c such that $0 \leq \alpha_{\mathbf{r},c} \leq 1$ and $\sum_{c=1}^{N_\mathcal{L}} \alpha_{\mathbf{r},c} = 1$. These labels are subject of learning itself, additionally to usual prototype adaptation. Yet, we remark that the class information used to distinguish the adaptation rules for correct and incorrect prototypes needed in (6.6) is no longer available now. Hence, in addition to an update rule for the fuzzy labels, we have to introduce a new methodology for appropriate prototype adaptation.

For this purpose we can use the material provided by SNPC. First, we can state that the loss boundary property (6.5) remains valid under the new condition. Further, the stochastic derivative of the cost function (6.4) according to the weights determines the prototype adaptation:

$$\frac{\partial lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)}{\partial \mathbf{w_r}} = \frac{u_\tau\left(\mathbf{r}|\mathbf{v}_k\right)}{2\tau^2} \cdot \left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}} - 1 + lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)\right) \cdot \frac{\partial d_{\mathbf{r}}}{\partial \mathbf{w_r}} \tag{6.11}$$

Parallel, the fuzzy labels $\alpha_{\mathbf{r},c_{\mathbf{v}_k}}$ can be optimized by gradient descent on the cost function, too. Taking the local cost one has

$$\triangle \alpha_{\mathbf{r},c_{\mathbf{v}_k}} = -\frac{\partial lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)}{\partial \alpha_{\mathbf{r},c_{\mathbf{v}_k}}} \tag{6.12}$$

with

$$\frac{\partial lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)}{\partial \alpha_{\mathbf{r},c_{\mathbf{v}_k}}} = -u_\tau\left(\mathbf{r}|\mathbf{v}_k\right) \tag{6.13}$$

and subsequent normalization. In accordance to the modified dynamic the window rule has still to be adapted appropriately. The rule can be derived in analogy to LVQ2.1 and SNPC, and is necessary for numerical stabilization of the adaptation process [83],[135]. For this purpose we consider in (6.7) the term

$$T = u_\tau\left(\mathbf{r}|\mathbf{v}_k\right)\left(\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) - lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)\right) \tag{6.14}$$

paying attention to the fact that now the $\alpha_{\mathbf{r},c_{\mathbf{v}_k}}$ are fuzzy. Using the Gaussian form (6.2) for $u_\tau(\mathbf{r}|\mathbf{v}_k)$ the term can be rewritten as

$$T = T_0 \cdot \Pi\left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) \tag{6.15}$$

with

$$\Pi\left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) = \frac{\exp\left(-\frac{d(\mathbf{v}_k,\mathbf{w}_\mathbf{r})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)} \tag{6.16}$$

and

$$T_0 = \left(lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)\left(1-lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)\right)-\alpha_{r,c_{v_k}}\left(1+\alpha_{r,c_{v_k}}\right)\right) \tag{6.17}$$

Obviously, $0 \leq lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)\left(1-lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)\right) \leq 0.25$ because $lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)$ fulfills the loss boundary property (6.5). Hence, we have $-2 \leq T_0 \leq 0.25$ using the fact that $\alpha_{r,c_{v_k}} \leq 1$. Now, a similar argumentation as in [135] can be applied: The absolute value of the factor T_0 has to be significantly different from zero to have a valuable contribution in the update rule. This yields the *window condition* $0 \ll |T_0|$, which can be obtained by balancing the local loss $lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)$ and the value of the assignment variable $\alpha_{r,c_{v_k}}$. Details on the derivation can be found in the Appendix A.

In general the FSNPC algorithm will be used as a standalone algorithm to obtain a model for classification of new data. Thereby the FSNPC algorithm will be applied on an initial (e.g. randomly) setting of codebook vectors and fuzzy labeling. Alternatively one can use the FSNPC approach as a post learn algorithm. In this case FSNPC will be applied on an initially learned codebook vector distribution (e.g. by SNG as introduced in [164]) with random labeling or alternatively by use of an approach as given in [11],[35]. In this article we focus on the first variant.

6.3 Relevance Learning in Fuzzified Self Adapted SNPC

In analogy to the learning approaches GRLVQ or SRNG we now consider the relevance learning idea for SNPC and FSNPC. The metric d_r is replaced by a parametrized one and the metric adjustment for optimum classification is studied. The respective adjusting of the relevance parameters λ is determined by the derivative $\frac{\partial lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)}{\partial\lambda}$ using the local cost (6.4). This leads to the update rule:

$$\frac{\partial lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)}{\partial\lambda_j} = \frac{\partial}{\partial\lambda_j}\left[\frac{\sum_{\mathbf{r}}\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_\mathbf{r})}{2\tau^2}\right)\cdot\left(\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right)\right)}{\sum_{\mathbf{r}'}\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}\right] \tag{6.18}$$

$$= -\frac{1}{2\tau^2}\sum_{\mathbf{r}}u_\tau(\mathbf{r}|\mathbf{v}_k)\cdot\frac{\partial d^\lambda_\mathbf{r}}{\partial\lambda_j}\cdot\left(\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-lc\left(\left(\mathbf{v}_k,c_{\mathbf{v}_k}\right),\mathbf{W}\right)\right)\right) \tag{6.19}$$

with subsequent normalization of the λ_k.

6.4 Localized Metric Adaptation in FSNPC and SNPC

Up to now FSNPC has been used with a global metric in the distance calculations. Now consider the case of localized metric adaptation for FSNPC. This can be done similarly as in localized GRLVQ. The respective adjusting of the relevance parameters λ is again determined by $\frac{\partial lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)}{\partial \lambda_{\mathbf{r}}}$ using the local cost (6.4), now with local metrics. In complete analogy one gets:

$$\frac{\partial lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)}{\partial \lambda_{rj}} = \frac{\partial}{\partial \lambda_{\bar{r}j}} \left[\frac{\sum_{\mathbf{r}} \exp\left(-\frac{d^{\lambda}(\mathbf{v}_k, \mathbf{w}_{\mathbf{r}})}{2\tau^2}\right) \cdot \left(\left(1 - \alpha_{\mathbf{r}, c_{\mathbf{v}_k}}\right)\right)}{\sum_{\mathbf{r}'} \exp\left(-\frac{d^{\lambda}(\mathbf{v}_k, \mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)} \right] \tag{6.20}$$

$$= -\frac{1}{2\tau^2} u_{\tau}\left(\bar{\mathbf{r}} | \mathbf{v}_k\right) \cdot \frac{\partial d^{\lambda}_{\bar{\mathbf{r}}}}{\partial \lambda_j} \cdot \left(\left(1 - \alpha_{\mathbf{r}, c_{\mathbf{v}_k}} - lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)\right)\right) \tag{6.21}$$

but now with subsequent normalization of the λ_k for each $\mathbf{w}_{\bar{\mathbf{r}}}$ separately.

6.5 Interpolated Voronoi tessellation

In the former non-fuzzy approaches, the winner determination has been done considering the closest prototype. This gives a (crisp) Voronoi tessellation of the input data space according to the nearest neighbor rule. If the prototype labels become fuzzy it is natural to extend the classical winner determination by not just measuring the distance in the feature space but also in the label space. To do so we normalize the distances d_i of a data point $\mathbf{v_i}$ such that the closest prototype gets a weight of one with decreasing weights for more distant prototypes inducing new variables $\mathbf{d}$. Now for each prototype the fuzzy labeling is considered and the classes $p_j^{\max}$ with maximum probability are determined. The fuzzy winner is finally determined as:

$$\Psi_{V \to \mathcal{A}} : \mathbf{v} \mapsto \mathbf{s}\left(\mathbf{v}\right) = \mathbf{argmax}_{\mathbf{r} \in A} \mathbf{d}\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}}\right) \cdot p_r^{\max}$$

This kind of Voronoi tessellation is beneficial if the underlying data analysis problem is fuzzy at the class boundaries ([19]).

6.6 Experiments

We now apply the FSNPC algorithm on a synthetic set of 1800 data points in two classes. Each class consists of 900 data points in two dimensions and the data is Gaussian distributed with different variances per data class and a small overlapping

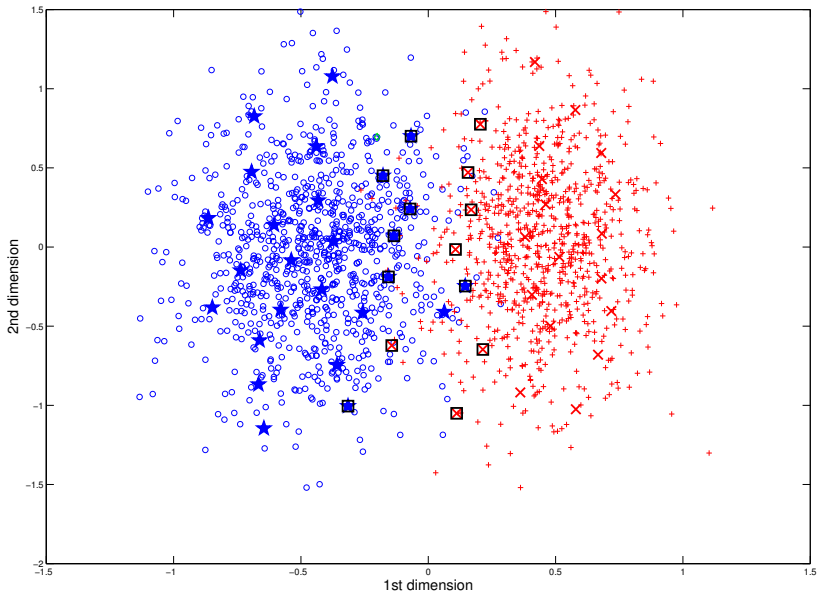


Figure 6.1: Plot of the overlapping synthetic data sets. Data of class 0 are given as circles together with respective codebook vectors as stars, data of class 1 are given by '+'-signs with their codebook vectors as $\times$. Codebook vectors without clear decision (labeling) are indicated by a surrounding black square.

region between the two data distributions. The obtained data space is shown in Figure (6.1).

We use the FSNPC as a standalone algorithm to learn a model for the codebook vector distribution and the fuzzy prototype labeling. We use 50 codebook vectors, i.e., 25 codebook vectors per class. The initial fuzzy labeling is randomly initialized at about 50% for each class per codebook vector. This leads to an initial prediction of around 46%. The FSNPC algorithm now optimizes in each step the codebook vector positions and label information. Because of the fuzzy property the codebook labels can change during optimization. Indeed the labeling becomes nearly perfect until the 50th complete step of FSNPC which leads to a prediction of 92%.

To assess the classification rate we assign prototypes with an at least 60% responsibility for a certain to this class. With this threshold we obtain a sufficiently good labeling after 300 complete steps. Note, that codebook vectors which are clearly located within the region of a data class show very pronounced fuzzy labels of about $80\% - 100\%$ for the correct class. Only codebook vectors close or in the overlapping class region are still undecided with fuzzy labels of approximately 50% for each class. It can be seen during training that the codebook vectors in the overlap region switch frequently their labeling. This indicates for the respective data points that the classification should be taken as an unknown class label. This behavior is shown in Figure (6.1).

Part III

Generalization theory and parameter optimization

Chapter 7

Generalization Theory for Learning Vector Quantization

An idea is always a generalization, and generalization is a property of thinking. To generalize means to think.
Georg Wilhelm Hegel

Learning machines such as the family of Learning Vector Quantizers with localized and variable metrics or (linear) kernel based approaches such as Support Vector Machines greatly increase the expressive power of learning machines while keeping the learning problem tractable. The increased flexibility, however, increases the risk of overfitting. Successful controlling the increased flexibility requires a sophisticated theory of generalization. Several learning theories exist that can be applied to this problem. The theory of Vapnik and Chervonenkis (VC) [155] is the historical appropriate theory to describe SVMs, but is going to be substituted by a recently introduced approach upon Rademacher and Gaussian Complexity measures which is also well suitable for Generalized Learning Vector Quantizers. In the following we give a short review on Generalization Theory focusing on Rademacher and Gaussian complexities which have been found to give direct bounds. Thereafter we describe the generalization ability of GRLVQ networks and how generalization bounds can also be obtained for GRLVQ networks with localized metric adaptation. One of the goals of the generalization bounds is to provide a model selection criterion, which can be used to determine quantities or parameters of the Learning machine which lead to good generalization.

7.1 Introduction to Generalization Theory

The generalization ability of a classifier refers to the expected error for data not used for training compared to the training error. There are various ways to formalize and prove the generalization ability of classifiers, such as the popular VC-theory [155] or, recently, argumentation based on Rademacher and Gaussian complexity [4]. Beside

these formal theories, empirical methods are popular such as crossvalidation which is easy to understand and shows good performance in practical applications ([77]). These simple approaches are often beneficial to evaluate the prediction capability of a determined classification model. Nevertheless cross validation is a posterior approach. The results obtained from crossvalidation approaches (e.g. leave one out, k-fold or random subset selection) give posterior estimations of the generalization error for a specific dataset, but no prior guaranty of the capability of a mechanism for PAC learning. Hence, a formal theory of generalization is needed to get a well founded mathematical formulation of the learnability. The VC-theory is widely applied for the generalization analysis of kernel methods, but it gives only raw generalization bounds [57]. The approach of Bartlett estimates the generalization bounds using Rademacher and Gaussian complexities which allow better estimations and are easier to handle especially in terms of GRLVQ networks. The main focus is to derive formal generalization bounds which limit the generalization error which can be expected by a learning machine with multiple parameters. To model the different kinds of classifier functions within a common theoretical framework the estimations of generalization bounds are based upon generic functions e.g. Gaussian functions. Thereby the obtained bounds of the estimated generalization error should be as close as possible to the real generalization error. The bounds can be used further to compare different classifier models[1] to obtain a classifier with a good generalization ability in terms of small error bounds and a small number of model parameters. Subsequently we focus on such a formal theoretic approach and show how it can be applied to GRLVQ networks.

7.1.1 Rademacher and Gaussian Complexities

We now resume the main ideas necessary for Rademacher and Gaussian Complexities as introduced by Bartlett et al. [4] which will be very helpful for the derivation of generalization boundaries for LVQ networks. Therein function classes that can be expressed as combinations of functions from basis classes are considered. It is shown how the Rademacher and Gaussian complexities of such a function class can be bounded in terms of the complexity of the basis classes.

Typically, such bounds take the form of a sum of two terms: some sample-based estimate of performance and a penalty term that is large for complex models. Such an example is given by [4] as an improvement of a classical result of Vapnik and Chervonenkis [155]:

Theorem 1 *Let F be a class of ± 1-valued functions defined on a set χ. Let P be a probability distribution on $\chi \times \pm 1$, and suppose that $(X_1, Y_1), \ldots, (X_m, Y_m)$ and (X, Y) are chosen independently according to P. Then, there is an absolute constant c such that for any integer m, with probability $1 - \delta$ over samples of length m, every*

[1] One may calculate multiple classifiers with different numbers of variables in the model or different structures.

$f \in F$ satisfies:

$$P(Y \neq f(X)) \leq \hat{P_m}(Y \neq f(X)) + c\sqrt{\frac{VCdim(F)}{m}}$$

where VCdim(F) denotes the Vapnik-Chervonenkis dimension of F,

$$\hat{P_m}(S) = \frac{1}{m}\sum_{i=1}^{m} 1_{S(X_i,Y_i)}$$

and 1_S is the indicator function of S

In this case, the sample-based estimate of performance, is the fraction of examples in the set of training samples that are misclassified by the function f on the one side and the complexity penalty term involving the VC-dimension of the class of functions in the other hand. It is natural to use such bounds for the model selection scheme, known as complexity regularization [4]. We choose the model class containing the function with the best upper bound on its error. Such a selection critically depends on how well the estimated error bounds match the true error [3]. As Kearns in [77] note it exists theoretically and experimental evidence that error bounds with a fixed constant penalty term (e.g. not depending on the training samples) cannot be universally effective. Hence a more complex functional penalty term is needed. Such a term can be derived by a function based generalization theory. The following definitions and terms are necessary basics to derive this penalty term in terms of Gaussian and Rademacher complexities.

Definition 7 *Let P be a probability distribution on a set χ and suppose that $X_1, \ldots, X_m$ are independent samples selected according to P. Let F be a class of functions mapping from χ to $\mathbb{R}$. Define the maximum discrepancy of F as the random variable*

$$\hat{D_m}(F) = \sup_{f \in F}\left(\frac{2}{m}\sum_{i=1}^{m/2} f(X_i) - \frac{2}{m}\sum_{i=m/2+1}^{m} f(X_i)\right)$$

Denote the expected maximum discrepancy of F by $D_m(F) = E\hat{D_m}(F)$.

Definition 8 *Define the random variable*

$$\hat{R_m}(F) = E\left[\sup_{f \in F}|\frac{2}{m}\sum_{i=1}^{m}\sigma_i f(X_i)||X_1, \ldots, X_m\right]$$

where $\sigma_1, \ldots, \sigma_m$ are independent uniform (± 1)-valued random variables. Then the Rademacher complexity of F is defined as

$$R_m(F) = E\hat{R_m}(F).$$

Definition 9 *Similarly, define the random variable*

$$\hat{G_m}(F) = E\left[\sup_{f\in F}|\frac{2}{m}\sum_{i=1}^{m} g_i f(X_i)||X_1,\ldots,X_m\right]$$

where $g_1,\ldots,g_m$ are independent Gaussian $N(0,1)$ random variables. The Gaussian complexity of F is $G_m(F) = E\hat{G_m}(F)$.

$\hat{D_m}(F)$ quantifies how much the behavior on half of the sample can be unrepresentative of the behavior on the other half, and both $R_m(F)$ and $G_m(F)$ quantify the extent to which some function in the class F can be correlated with a noise sequence of length m.

We now briefly review some theorems taken from [4] which are needed in the following derivations.

Theorem 2 *Let P be a probabilistic distribution on $\chi \times \pm 1$ and let F be a set of real-valued functions defined on χ, with $\sup\{|f(x))| : f \in F\}$ finite for all $x \in \chi$. Suppose that $L : \mathbb{R} \to [0,1]$ satisfies $L(\alpha) \geq 1(\alpha \leq 0)$ and is Lipschitz continuous with constant C. Then with probability at least $1-\delta$ with respect to training samples $(X_i, Y_i)_{i=1}^m$ drawn according to P^m each function in F satisfies*

$$P(Yf(X) \leq 0) \leq \hat{E_m}L(Yf(x)) + 2CR_m(F) + \sqrt{\frac{\ln(2/\delta)}{2m}}$$

An important property of Rademacher complexity is that it can be estimated from a single sample and from a single realization of the Rademacher variables.

Theorem 3 *Let F be a class of functions mapping to $[-1,1]$. For any integer m,*

$$P\left\{\left|R_m(F) - \frac{2}{m}\sup_{f\in F}\left|\sum_{i=1}^{m}\sigma_i f(X_i)\right|\right| \geq \epsilon\right\} \leq 2\exp\left(\frac{-\epsilon^2 m}{8}\right),$$

and

$$P\left\{\left|R_m(F) - \hat{R_m}(F)\right| \geq \epsilon\right\} \leq 2\exp\left(\frac{-\epsilon^2 m}{8}\right),$$

Thus, it seems that the estimation of $R_m(F)$ and $G_m(F)$ is particularly convenient. However, the computation involves an optimization over the class F, which is hard for interesting function classes. Therefore Bartlett et al. gave further structural results and showed how such large classes can be expressed as combinations of functions from simpler classes. For the boolean combination of functions the following theorem was given in [4]

Theorem 4 *For a fixed boolean function $g : \{\pm 1\}^k \to \{\pm 1\}$ and classes $F_1, \ldots, F_k$ of ± 1-valued functions,*

$$G_m(g(F_1, \ldots, F_k)) \leq 2 \sum_{j=1}^{k} G_m(F_j)$$

We close this chapter with the Lemma for the upper bound estimation of Gaussian and Rademacher complexities for kernel functions given in [4]

Lemma 1 *Suppose that $k : \chi \times \chi \to \mathbb{R}$ is a kernel, and let $\chi_1, \ldots, \chi_m$ be random elements of χ. Then for the class F with elements $f(x) = \sum_{i=1}^{m} \alpha_i k(X_i, x)$ the bounds are*

$$\begin{aligned} \hat{G_m}(F) &\leq \frac{2B}{m} \sqrt{\sum_{i=1}^{m} k(X_i, X_i)} \\ \hat{R_m}(F) &\leq \frac{2B}{m} \sqrt{\sum_{i=1}^{m} k(X_i, X_i)} \end{aligned}$$

7.2 Generalization ability of Generalized Learning Vector Quantization

In advanced LVQ networks a lot of structural parameters comes into play, defining the learning machine. These structural parameters such as the number of prototypes, the upper bound of the prototype length and the number of items in the training set, affect the generalization ability of the LVQ network and can be used for model selection. To formalize this selection and to give a well estimation of the generic empirical risk a boundary estimation based upon ideas suggested by Bartlett [4] can be taken.

7.2.1 Generalization bounds for GRLVQ

We now give a review of the recently derived generalization bounds for GRLVQ networks [57], which is followed by a generalization on GRLVQ networks with *local* metric adaptation. We start with generalization bounds for adaptive metrics based on Rademacher and Gaussian Complexities as introduced in [4]. The bound depends on the margin of the classifier and is independent of the dimensionality of the data. It holds for classifiers based on the Euclidean metric extended by adaptive relevance factors. In particular the result holds for relevance learning vector quantization and GRLVQ with scaled Euclidean metric.

LVQ networks show excellent generalization for unseen data as well as for high-dimensional inputs. A theoretical explanation for this fact has recently been established [33] with generalization bounds of LVQ classifiers which do not depend on the input dimensionality but only on the so called hypothesis margin. However, the theoretical large margin bound on the generalization ability of LVQ as derived in [33] does not transfer to the case of an adaptive metric. It has been recently shown (in [33], Lemma 1) that the term

$$\frac{||\mathbf{v}_k - \mathbf{w}_{r^-}|| - ||\mathbf{v}_k - \mathbf{w}_{r^+}||}{2} \tag{7.1}$$

constitutes the so-called hypothesis margin of a prototype-based classifier according to the winner-takes-all rule (3.1).

The generalization bound derived in [33] holds for all prototype based classifiers for which the classification rule is given by (3.1). The bound does not longer hold for prototype-based classifiers with classification rule (5.1) exhibiting changing parameter λ during training. In the following a large margin generalization bound for the latter case is given as derived in [57].

Here, we consider the situation of binary classification problems, i.e. only two classes are given. We assume the classes are labeled 1 and -1. Assume some (unknown) probability measure P is fixed on $\mathbb{R}^n \times \{-1, 1\}$. Training samples $(\mathbf{v_k}, c_{\mathbf{v_k}})$ are drawn independently and identically distributed (i.i.d. for short) from $\mathbb{R}^n \times \{-1, 1\}$. P^m refers to the product of P if m examples $(\mathbf{v_1}, c_{\mathbf{v_1}}), \ldots, (\mathbf{v_m}, c_{\mathbf{v_m}})$ are chosen. An unknown regularity should be learned by a GRLVQ-network or some other prototype-based classifiers with a (parametric) diagonal metric (compare chapter 5). The classifier is characterized by its set of prototypes $\mathbf{w}_1, \ldots, \mathbf{w}_{N_W}$ in $\mathbb{R}^n$ (with N_W denoting the number of prototypes) and the relevance terms $\lambda_1, \ldots, \lambda_n$ which describe the weighted metric. The function computed by the classifier is given by the winner-takes-all rule (5.1). Let F be the set of all GRLVQ network functions such that

$$F = \{f : \mathbb{R}^n \to \{-1, 1\} | f \text{ represented by (5.1) for some } \mathbf{w}_1, \ldots, \mathbf{w}_{N_W}, \lambda \in \mathbb{R}^n\}$$

the class of functions which can be computed by such a network. The goal of learning is to find a function $f \in F$ such that the probability

$$E_P(f) := P(c \neq f(\mathbf{v_k}))$$

is as small as possible. Since the underlying regularity P is not known and only examples $(\mathbf{v_k}, c_{\mathbf{v_k}})$ for this regularity are available, training tries to minimize the empirical training error:

$$\hat{E}_m(f) := \frac{1}{m} \sum_{k=1}^{m} \mathbf{1}_{c_k \neq f(\mathbf{v_k})} \tag{7.2}$$

whereby $\mathbf{1}_{c_k \neq f(\mathbf{v_k})}$ indicates whether $\mathbf{v_k}$ is mapped to the desired class c_k or not. Generalization means that $\hat{E}_m(f)$ is representative for $E(f)$ with high probability if the

examples are chosen according to P^m such that optimization of the empirical training error will eventually approximate the underlying regularity. Due to the chosen cost function, GRLVQ minimizes the training error and, in addition, also optimizes the margin of the classifier during training. Given a point $\mathbf{v_k}$ with desired output c, we define the margin as the value

$$M_f(\mathbf{v_k}, c) := -d^{\lambda}_{r^+} + d^{\lambda}_{r^-} \tag{7.3}$$

If the vector $\mathbf{v_k}$ is classified incorrectly, $M_f(\mathbf{v_k}, c)$ is negative. Otherwise, $\mathbf{v_k}$ is classified correctly with 'security' or margin $M_f(\mathbf{v_k}, c)$. Due to the choice of the cost function of GRLVQ which involves this term within the denominator, GRLVQ aims at maximizing this margin. Following the approach in [4] we define the following loss function

$$L : \mathbb{R} \to \mathbb{R}, t \to \begin{cases} 1 & \text{if } t \leq 0 \\ 1 - \frac{t}{\rho} & \text{if } 0 < t \leq \rho \\ 0 & \text{otherwise} \end{cases} \tag{7.4}$$

for some fixed $\rho > 0$.
The term

$$\hat{E}^L_m(f) := \frac{1}{m} \sum_k^m L(M_f(\mathbf{v_k}, c))$$

then accumulates the number of errors made by the f and, in addition, also punishes all correctly classified points if their margin is smaller than ρ. It was shown in [57] that this modified empirical error, which also includes the margin, is representative for the true error with high probability, whereby a dimensionality independent bound is obtained. Let the support of the probability measure P be bounded, i.e. for all data points $\mathbf{v}$ the inequality

$$||\mathbf{v}|| \leq B$$

holds for some $B > 0, ||\cdot||$ denoting the standard Euclidean metric. In addition, all prototypes w are restricted by

$$||\mathbf{w}|| \leq B.$$

According to theorem (2) we can estimate for all $f \in F$ with probability at least $1 - \frac{\sigma}{2}$

$$E_P(f) \leq \hat{E}^L_m(f) + \frac{2K}{\rho} \cdot G_m(F) + \sqrt{\frac{\ln(4/\sigma)}{2m}}$$

whereby K is a universal positive constant and $G_m(F)$ is the Gaussian complexity of the following considered function class. Here the classification by the winner-takes-all rule (5.1) can be reformulated as fixed Boolean formula over terms of the form $d^{\lambda}_{r^k} - d^{\lambda}_{r^l}$ with $d^{\lambda}_{r^k}$ and $d^{\lambda}_{r^l}$ constituting the weighted squared Euclidean distance of a given input $\mathbf{v}$ to two different prototypes $\mathbf{w}_k$ and $\mathbf{w}_l$ with different labels. Note that the number

of such terms is upper bounded by $N_W \cdot (N_W - 1)/2$ if N_W prototypes are available within the classifier. According to theorem (4) one founds

$$G_m(F) \leq N_W \cdot (N_W - 1) \cdot G_m(F_{kl})$$

whereby F_{kl} denotes the restricted class of classifiers which can be implemented with only two prototypes w_k and w_l with different class labels. Denote by Λ the diagonal matrix of relevance factors

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & 0 \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & \dots & \dots & \lambda_n \end{pmatrix} \tag{7.5}$$

Then we find

$$\begin{aligned} & d^{\lambda}_{r^k} - d^{\lambda}_{r^l} \leq 0 \\ \leftrightarrow \quad & (\mathbf{v} - \mathbf{w}_{r^k})^{\top} \cdot \Lambda \cdot (\mathbf{v} - \mathbf{w}_{r^k}) - (\mathbf{v} - \mathbf{w}_{r^l})^{\top} \cdot \Lambda \cdot (\mathbf{v} - \mathbf{w}_{r^l}) \leq 0 \\ \leftrightarrow \quad & 2 \cdot (\Lambda \cdot \mathbf{w}_{r^k} - \Lambda \mathbf{w}_{r^l})^{\top} \mathbf{v_k} + (\mathbf{w}_{r^l})^{\top} \cdot \Lambda \cdot \mathbf{w}_{r^l} - (\mathbf{w}_{r^k})^{\top} \cdot \Lambda \cdot \mathbf{w}_{r^l} \geq 0 \end{aligned}$$

Hence we can embed F_{kl} into the class of functions implemented by a simple perceptron or linear classifier. Since $||\mathbf{v}|| \leq B$, the length of inputs to the linear classifier can be restricted by $B + 1$ (including the bias term). Since all prototypes $\mathbf{w}$ are restricted by $||\mathbf{w}|| \leq B$ and the relevance factors λ_i add up to 1, the size of the weights of the linear classifier is restricted by $6B$. The empirical Gaussian complexity of this class of linear classifiers can be estimated according to lemma (1) by

$$\frac{12 \cdot B \cdot (B+1) \cdot \sqrt{m}}{m}$$

Since the empirical Gaussian complexity and the Gaussian complexity differ by more than ϵ with probability at most $2 \cdot \exp(-\epsilon^2 m/8)$ according to theorem (3) we can estimate

$$E_P(f) \leq \hat{E}^L_m(f) + \frac{2K \cdot N_W(N_W - 1) \cdot 12 \cdot B \cdot (B+1)}{\rho \cdot \sqrt{m}} \tag{7.6}$$

$$+ \left(1 + \frac{8K \cdot N_W(N_W - 1)}{\rho}\right) \sqrt{\frac{\ln 4/\delta}{2m}} \tag{7.7}$$

$$\leq \hat{E}^L_m(f) + \frac{1}{\rho \cdot \sqrt{m}} \cdot \sqrt{\ln 1/\delta} \cdot O(K {N_W}^2 B^2) \tag{7.8}$$

with probability at least $1-\delta$. This term limits the generalization error for all classifiers of the form (5.1) with possibly adaptive relevance terms λ. In particular, it holds for the functions obtained by the algorithms RLVQ and GRLVQ. In addition, the bound indicates that GRLVQ includes the objective of structural risk minimization during training because a term, which describes the margin M_f (7.3), is directly contained in the cost function of GRLVQ.

7.2.2 Generalization bounds for GRLVQ with local Metric Adaptation

We now derive generalization bounds for localized GRLVQ (LGRLVQ) which is similar to the procedure of the previous section. The classifier is characterized by its set of prototypes $\mathbf{w_1}, \ldots \mathbf{w_{N_W}} \in \mathbb{R}^n$ and the respective relevance factors $\lambda_\mathbf{1}, \ldots, \lambda_{\mathbf{N_W}}$ which describe the local weighted metrics. The function computed by the classifier is given by the winner-takes all rule defined in (5.22). Denote by:

$$F = \{f : \mathbb{R}^n \to \{-1, 1\} | f \text{ is given by (5.22) for some } \mathbf{w}_1, \ldots, \mathbf{w}_{N_W}, \lambda_\mathbf{1}, \ldots \lambda_{\mathbf{N_W}} \in \mathbb{R}^n\}$$

the class of functions which can be computed by such a network. The goal of the learning is to find a function $f \in F$ such that the probability

$$E_P(f) := P(c \neq f(\mathbf{v_k}))$$

is minimum. Since the underlying regularity P is not known and only examples $(\mathbf{v_k}, c_{\mathbf{v_k}})$ for this regularity are available, training tries to minimize the empirical training error (7.2). Due to the chosen cost function, LGRLVQ minimizes the training error and, in addition, also optimizes the margin of the classifier during training. Given a point $\mathbf{v_k}$ with desired output c, we analogously define the margin with *local* metric adaptation as the value

$$M_f(\mathbf{v_k}, c) := -d_{r^+}^{\lambda^+} + d_{r^-}^{\lambda^-} \tag{7.9}$$

The vector $\mathbf{v_k}$ is classified incorrectly if $M_f(\mathbf{v_k}, c)$ is negative. Otherwise, $\mathbf{v_k}$ is classified correctly with 'security' or margin $M_f(\mathbf{v_k}, c)$. Due to the choice of the cost function of LGRLVQ which involves this term within the denominator, LGRLVQ aims at maximizing this margin.

We now go on equivalently to the boundary estimation as for GRLVQ and realize that the classification by the winner-takes-all rule with now local metric adaptation can be reformulated again as fixed Boolean formula over terms of the form $d_{r^k}^{\lambda^k} - d_{r^l}^{\lambda^l}$ with $d_{r^k}^{\lambda^k}$ and $d_{r^l}^{\lambda^l}$ constituting the local weighted squared Euclidean distance of a given input $\mathbf{v}$ to two different prototypes $\mathbf{w}_k$ and $\mathbf{w}_l$ with different labels. As before, note that the number of such terms is upper bounded by $N_W \cdot (N_W - 1)/2$ if N_W prototypes are available within the classifier. According to theorem (4) we find

$$G_m(F) \leq N_W \cdot (N_W - 1) \cdot G_m(F_{kl})$$

whereby F_{kl} denotes the restricted class of classifiers which can be implemented with only two prototypes w_k and w_l with different class labels. Denote by Λ^k the diagonal matrix of relevance factors λ_i^k. Then we find for a fixed k and i

$$\begin{aligned}
& d_{r^k}^{\lambda^k} - d_{r^l}^{\lambda^l} \leq 0 \\
\leftrightarrow \quad & (\mathbf{v} - \mathbf{w}_{r^k})^\top \cdot \Lambda^k \cdot (\mathbf{v} - \mathbf{w}_{r^k}) - (\mathbf{v} - \mathbf{w}_{r^l})^\top \cdot \Lambda^l \cdot (\mathbf{v} - \mathbf{w}_{r^l}) \leq 0 \\
\leftrightarrow \quad & \mathbf{v}^\top \cdot \Lambda^k \cdot \mathbf{v} - \mathbf{v}^\top \cdot \Lambda^l \cdot \mathbf{v} - 2 \cdot (\Lambda^k \cdot \mathbf{w}_{r^k} - \Lambda^l \mathbf{w}_{r^l})^\top \mathbf{v_k} + (\mathbf{w}_{r^l})^\top \cdot \Lambda^k \cdot \mathbf{w}_{r^l} \\
& -(\mathbf{w}_{r^k})^\top \cdot \Lambda^l \cdot \mathbf{w}_{r^l} \leq 0
\end{aligned}$$

Hence, every function from F_{kl} can be written as the sum of a function from the set $F_k = \{\mathbf{v} \to \mathbf{v}^\top \cdot \Lambda^k \cdot \mathbf{v}\}$, a function from the set - F_l, and a function implemented by a simple perceptron or linear classifier. According to [4] theorem 12, it holds

$$G_m(c \cdot F) = c \cdot G_m(F) \text{ and } G_m(\sum_i F_i) \leq \ln m \sum_i G_m(F_i).$$

Thus it is sufficient to independently estimate the Gaussian complexity of linear and quadratic functions of this form. As before, for linear functions the estimation follows immediately: since $||\mathbf{v}|| \leq B$, the length of inputs to the linear classifier can be restricted by $B + 1$ (including the bias term). Since all prototypes $\mathbf{w}$ are restricted by $||\mathbf{w}|| \leq B$ and the relevance factors add up to 1, the size of the weights of the linear classifier is restricted by $4B + 2B^2$. The empirical Gaussian complexity of this class of linear classifiers can be estimated according to lemma (1) by

$$\frac{4 \cdot B \cdot (B+1) \cdot (B+2) \cdot \sqrt{m}}{m}$$

The empirical Gaussian complexity and the Gaussian complexity differ by more than ϵ with probability at most $2 \cdot \exp(-\epsilon^2 m/8)$ according to theorem (3).

Since we can interpret the mapping $(\mathbf{v} \to (\mathbf{v_1^2}, \ldots, \mathbf{v_n^2}))$ as feature map of a kernel, an estimation of the Gaussian complexity for the considered quadratic functions is also possible: for $\mathbf{v} \to \sum_i \lambda_i^k \mathbf{v_i^2}$ with $||\lambda_k|| \leq 1$ we can estimate the empirical Gaussian complexity using lemma (1) with $||\mathbf{v}|| \leq B$ by

$$\frac{2 \cdot B^2 \cdot \sqrt{m}}{m}$$

Thus, the overall error bound becomes

$$E_P(f) \leq \hat{E}_m^L(f) + \frac{4K \cdot N_W(N_W - 1)(2B(B+1)(B+2) + B^2) \ln m}{\rho \cdot \sqrt{m}} \tag{7.10}$$

$$+ \left(1 + \frac{8K \cdot N_W(N_W - 1) \cdot \ln m}{\rho}\right) \sqrt{\frac{\ln 4/\delta}{2m}} \tag{7.11}$$

$$\leq \hat{E}_m^L(f) + \frac{\ln m}{\rho \cdot \sqrt{m}} \cdot \sqrt{\ln 1/\delta} \cdot O(K {N_W}^2 B^3) \tag{7.12}$$

with probability at least $1 - \delta$. This term limits the generalization error for all classifiers of the form (5.22) with adaptive metric if only two classes are dealt with and inputs and weights are restricted by B. Note that this bound is independent of the dimensionality n of the data. It scales inversely to the margin ρ, i.e. the larger the margin the better the generalization ability. This bound indicates that LGRLVQ includes the objective of structural risk minimization during training because the terms $M_f(\mathbf{v_k}, c)$ which characterize the margin are directly contained in the cost function of LGRLVQ. Naturally, only the extremal margin values need to be limited and thus a restriction of the respective update to extremal pairs of prototypes would suffice. Thus, this argument even proposes schemes for active data selection if a fixed and static pattern set is available for training to speed the algorithm and improve its convergence.

Chapter 8

Heuristics for optimized Design and Learning

First, have a definite, clear practical ideal; a goal, an objective. Second, have the necessary means to achieve your ends; wisdom, money, materials, and methods. Third, adjust all your means to that end.
Aristotle

We now derive some generic methods to improve the computational performance of LVQ networks. Thereby we incorporate the idea of margin optimization from the previous chapter to support supervised active learning within GLVQ networks and methods for the dynamic adaptation of the network architecture to the data distribution under the considered classification task.

8.1 Active Learning Strategies

We now extend the local prototype-based learning model of Generalized LVQ networks by active learning, which gives the learner the capability to select training samples and thereby increase speed and accuracy of the model. Our algorithm is based on the idea of selecting a query on the borderline of the actual classification. This can be done by considering margins in an extension of learning vector quantization based on an appropriate cost function. The performance of the query algorithm is demonstrated on artificial data and real life data taken from clinical proteomic profiling studies.

In supervised learning, we frequently are interested in training a classifier based on given vectors and corresponding class labels such that the underlying (unknown) target distribution is well estimated. Whereas traditional approaches usually adapt the model according to all available and randomly sampled training data, the field of active learning restricts to only few actively selected samples. This method avoids the shortcoming of traditional approaches that the average amount of new information per sample decreases during learning and that additional data from some regions are

basically redundant. Further, it accounts for the phenomenon which is increasingly common e.g. in bioinformatics or web search that unlabeled data are abundant whereas reliable labeling is costly. Interestingly, different variants of active and query based learning have been proposed quite early for neural models [7, 28, 45, 79, 138] and the discussion is accompanied by literature pointing out its cognitive relevance ([119]).

In query algorithms proposed so far, samples are chosen according to some heuristic [7, 79, 70] or in a principled way by optimizing an objective function such as the expected information gain of a query [138, 45], or the model uncertainty [28, 49, 10]. Several interesting new models which are accompanied by crisp mathematical guaranties have been proposed recently [25, 34]. A common feature of these query algorithms, however, is that they have been applied to global learning algorithms. Only a few approaches incorporate active strategies into local learning such as [65] where a heuristic query strategy for simple vector quantization is proposed. Here we include active learning into GRLVQ and variants.

As shown in Chapter 7 Generalized relevance learning vector quantization can be accompanied by a large margin generalization bound [58], which is directly connected to the cost function of the algorithm and which opens the way towards active learning strategies, as we will discuss in the following.

This generalization bounds in terms of the margin proposes an elegant scheme to transfer margin based active learning to local learners. Margin based sample selection has been presented e.g. in the context of SVM in [151, 23, 133, 102]. Obviously, the generalization ability of the GRLVQ algorithm does only depend on the points with too small margin (7.3). Thus, only the extremal margin values need to be limited and a restriction of the respective update to extremal pairs of prototypes would be sufficient. This argument proposes schemes for active data selection if a fixed and static pattern set is available: We fix a monotonically decreasing non-negative function $L^c : \mathbb{R} \to \mathbb{R}$ and actively select training points from a given sample, in analogy to e.g. [102], based on the probability $L^c(M_f(\mathbf{v}, c_\mathbf{v}))$ for sample $\mathbf{v}$. Thereby, different realizations are relevant:

1. $L^c(t) = 1$ if $t \leq \rho$, otherwise, it is 0. That means, all samples with margin smaller than ρ are selected (Threshold strategy).

2. $L^c(t) = 1$ for $t > 0$ and $L^c(t) \sim |t|^\alpha$, otherwise, i.e. the size of the margin determines the probability of $\mathbf{v}$ being chosen annealed by a parameter α (Probabilistic strategy).

Both strategies focus on the samples which are not yet sufficiently represented in the model. Therefore, they directly aim at an improvement of the generalization bound (7.6,7.10). Strategy (1) allows an adaptation of the margin parameter ρ during training in accordance to the confidence of the model in analogy to the recent proposal [102] for SVM: for each codebook vector $\mathbf{w_r} \in W$ we introduce a new parameter α_r measuring the mean distance of data points in its receptive field (3.3). This parameter can be easily computed during training as a moving average with no extra costs. We choose

ρ_r locally as $\rho_r = 2 \cdot \alpha_r$. Thus, points with a margin which compares favorable to the size of the receptive fields are already represented with sufficient security and, hence, they are abandoned. For strategy (2), a confidence depending on the distance to the closest correct prototype and the overall classification accuracy can be introduced in a similar way. Doing this the normalized margin is taken as a probability measure for data selection.

We would like to mention that, so far, we have restricted active selection strategies to samples where all labels are known beforehand, because the closets correct and wrong prototype have to be determined in (7.3). This setting allows to improve the training speed and performance of batch training. If data are initially unlabeled and queries can be asked for a subset of the data, we can extend these strategies in an obvious way towards this setting: in this case, the margin (7.3) is given by the closest two prototypes which possess a different class label, whereby the (unknown) class label of the sample point has no influence. $L^c(t)$ is substituted by $L^c(|t|)$.

8.2 Dynamic Growing in Learning Vector Quantization

Prototype based classifiers gain increasing interest in bioinformatic applications ([125, 132, 172]). To obtain reliable results for the underlying problem a suitable parametrization is necessary. It has been reported by a number of researchers, that one disadvantage of neural-network models is the fact, that the network structure had to be specified in advance. This is generally not possible in an optimal way since a necessary piece of information, the probability distribution of the input signals, is usually not available. The choice of an unsuitable network structure, however, can badly degrade network performance as shown by Fritzke [47]. One important problem in modeling of prototype based networks is the adequate number of prototypes. Different approaches for self adaptive prototype networks has been proposed so far ([162, 48, 144, 115]) to overcome this problem. In the following we focus on Generalized Learning Vector Quantizers (GLVQ) as introduced in [122] - and derivatives e.g. [27] which allow for easy derivation of margin based measurements to control growing or shrinking strategies for adaptation of the number of prototypes. Dynamic adaptation becomes especially important if a useful initial setup of the network structure is hard to determine. In bioinformatic applications the complexity of the underlying data is hard to determine and a non expert usage of the algorithm requires for automatic network adaptation. This becomes immediately obvious considering highly multi modal data such as the checker board data set. But also in the light of life-long-learning, approaches for the dynamic adaptation of networks structures are needed and have already been studied e.g. by Fritzke [46] using a utilization factor or within the Adaptive resonance theory developed by Carpenter and Grossberg [24]. The presented method supports growing and shrinking strategies and can be initialized by the most simple

case giving one prototype for each class.

The dynamic adaptation of GLVQ networks aims on automatically determination of a problem specific network structure through a growth and shrinking process. Thereby we initialize the network with one prototype for each class followed by a learning and dynamic adaptation procedure. The former is realized by the well known learning dynamic of SATO & YAMADA [122] or derivatives and the second step incorporating ideas from FRITZKE [47, 48] modified to make special use of the margin optimization induced by GLVQ networks with good generalization bounds. In the dynamic GLVQ we adapt the concept of resource [47]. Thereby every prototype has a local resource variable which contains a counter indicating an error measure for the receptive field represented by the prototype. Every input signal causes an update of the resource variable of the best matching unit and the closest incorrect unit. Thereby the resource r_i for winner w_i and the resource t_j (life-time counter) for the next incorrect prototype w_j is adapted as follows:

$$\Delta r_{i,j} = \frac{d^{\lambda}_{\mathbf{r}_+} - d^{\lambda}_{\mathbf{r}_-}}{d^{\lambda}_{\mathbf{r}_+} + d^{\lambda}_{\mathbf{r}_-}} \tag{8.1}$$

The margin $M_f(\mathbf{v}, c_{\mathbf{v}}) = -d^{\lambda}_{\mathbf{r}_+} + d^{\lambda}_{\mathbf{r}_-}$ incorporated in the above term allows specific evaluation of the prototype performance. Thereby classifications with a positive margin determines correct classifications. Hence the size of the margin is directly related to the update of $r_{i,j}$ and gives over time a reliable performance measure for a prototype. In addition we add a life-time counter t to each prototype which is increased at each cycle. The resource parameter r is reset after a full cycle and moving average of r is stored in $\bar{r}$ overall cycles. Finally the resource of each prototype consists of the parameter vector $\wp = (r, \bar{r}, t)$.

8.2.1 Insertion of prototypes

Always after a constant number of adaptation steps (e.g. 10 cycle) a new prototype can be inserted. For this purpose a global resource accounting for misclassified training samples is evaluated and a random point is selected therefrom. The insertion is applied as long a predefined threshold of accuracy is not reached (e.g. 90%).

8.2.2 Deletion of prototypes

By comparing the fraction of all input signals which a specific prototype has received and the volume of its Voronoi region one can derive a local *estimate of the probability density* of the input vector [47]. Those prototypes, whose reference vectors fall into regions of the input vector space with a very low probability density, are regarded as *superfluous* and are removed. This can be simplified by evaluating the resource of each prototype. Prototypes with a low resource value r have very sparse receptive fields or

are located invalid with respect to neighbored regions. A corresponding threshold indicating *superfluous*-prototypes has to be chosen problem dependent by pre-knowledge about the smallest sub clusters of a class or experimental preevaluation. In addition the age of the prototype is kept in the prototype resource t. By definition of a *minimal survival* threshold one can force to keep prototypes in the model for at least a specific number of cycles. This is useful to support the learning dynamic of the algorithm. In our experiments this threshold is chosen by 20 cycles. Further we incorporated the moving average $\bar{r}$ for the resource value which has to be kept above a threshold to ensure the prototype is kept. This is especially useful at the beginning of the learning procedure and helps to keep the network stable until at least a raw ordering is obtained. This parameter can be omitted if the network is already prelearned by some alternative method. The result is a problem-specific network structure potentially modelling a given probability distribution.

8.3 Experiments and Results

We now compare the SRNG and SNPC with randomly selected samples with the SRNG, SNPC using the proposed active learning query strategies on artificial and clinical datasets taken from proteomic profiling studies. The real life data are the already introduced WDBC data set and the Proteom$_2$ data set. In addition the Checkerboard data are used and a spiral data set. The later one is used, to illustrate the differences between the different strategies during learning of the codebook vector positions. A simple spiral data set has been created and applied using the SRNG algorithm. The spiral data are generated in a similar way as the data shown in [65] and the set consists of 5238 data points in two dimensions.

For classification, we use 6 prototypes for the WDBC data, 100 prototypes for the well separable Checkerboard data set as given in [62], 9 prototypes for the Proteom$_2$ data set. Parameter settings for SRNG can be resumed as follows: learn rate correct prototype: 0.01, learn rate incorrect prototype: 0.001 and learning rate for λ: 0.01. The neighborhood range is given by $\#\mathbf{W}/2$. For SNPC the same settings as for SRNG are used with the additional parameter window threshold: 0.05 and width $\sigma = 2.5$ for the Gaussian kernel. Learning rates are annealed by an exponential decay. All data have been processed using a 10-fold cross-validation procedure. Results are calculated using the SNPC, SNPC-R and SNG, SRNG. SNG and SRNG are used instead of GLVQ or GRLVQ as improved version of the former using neighborhood cooperation.

We now compare both prototype classifiers using randomly selected samples with its counterparts using the proposed query strategies. The classification results are given in Tab. 8.1 and Tab. 8.3 without metric adaptation and in Tab. 8.2 and Tab. 8.3 with relevance learning respectively. Features of all data sets have been normalized. First we upper bounded the data set by 1.0 and subsequently data are transformed such that we end with zero mean and variance 1.0.

We applied the training algorithms using the different query strategies as intro-

duced above. The results for recognition and prediction rates using SRNG are shown in Tab. 8.2[1] and for SNPC in 8.3 respectively.

	SNG		$SNG_{\text{active strategy 1}}$			$SNG_{\text{active strategy 2}}$		
	Rec.	Pred.	Rec.	Pred.	Rel. #Q	Rec.	Pred.	Rel. #Q
WDBC	95%	95%	94%	94%	38%	93%	92%	9%
$Proteom_2$	76%	83%	76%	77%	48%	76%	85%	15%
Checker	72%	67%	98%	97%	31%	99%	96%	5%

Table 8.1: Classification accuracies for cancer and checkerboard data sets using SNG. The prediction accuracies are taken from a 10-fold crossvalidation and show a reliable good prediction of data which belong to the WDBC data as well as for the $Proteom_2$ data set. The Checker data are not as well modeled, this is due to the fixed upper limit of cycles (1000) longer runtimes lead for this data to a nearly perfect separation.

	SRNG		$SRNG_{\text{active strategy 1}}$			$SRNG_{\text{active strategy 2}}$		
	Rec.	Pred.	Rec.	Pred.	Rel. #Q	Rec.	Pred.	Rel. #Q
WDBC	95%	94%	95%	94%	29%	97%	97%	7%
$Proteom_2$	82%	88%	92%	87%	31%	97%	93%	5%

Table 8.2: Classification accuracies for cancer data sets using SRNG. Data characteristics are as given before. A reliable good prediction of data which belong to the WDBC data as well as for the $Proteom_2$ data set can be seen. One clearly observes an improved modeling capability by use of relevance learning and an related addition decrease in the number of queries.

[1]The relative number of queries is calculated with respect to the maximal number of queries possible up to convergence of SRNG using the corresponding query strategy. The upper limit of cycles has been fixed to 1000.

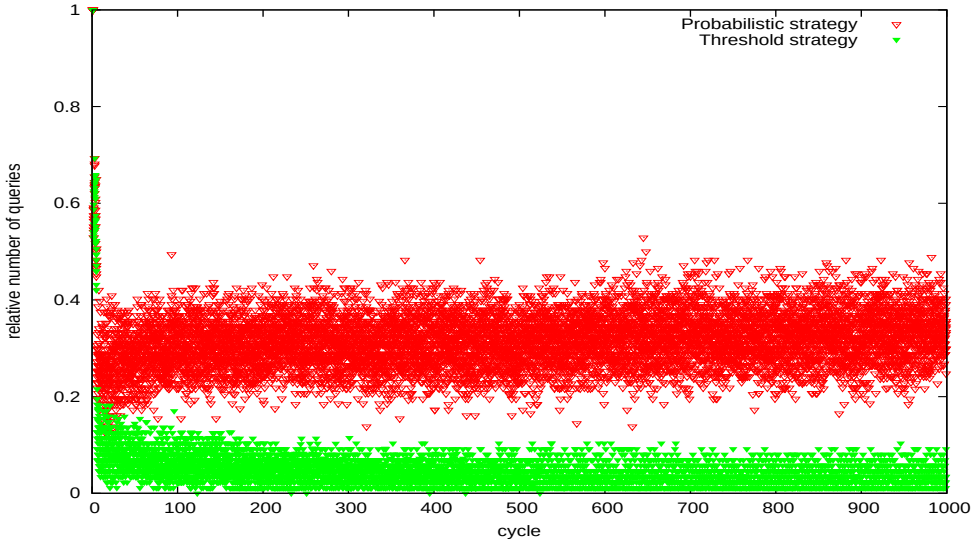


Figure 8.1: Number of queries in % using active strategy 1 (Threshold) and 2 (Probabilistic) executed by the SRNG algorithm on the Proteom$_2$ data set.

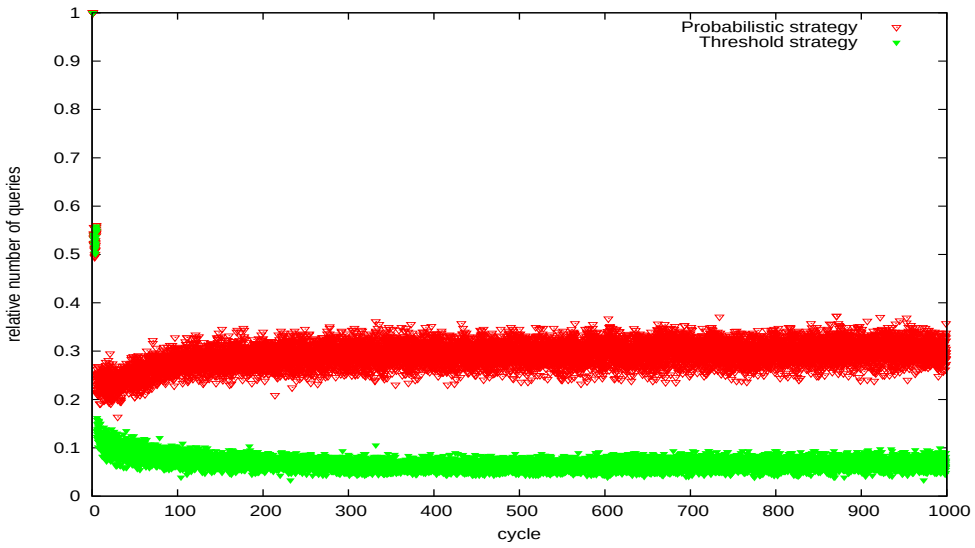


Figure 8.2: Number of queries in % using active strategy 1 (Threshold) and 2 (Probabilistic) executed by the SRNG algorithm on the WDBC data set.

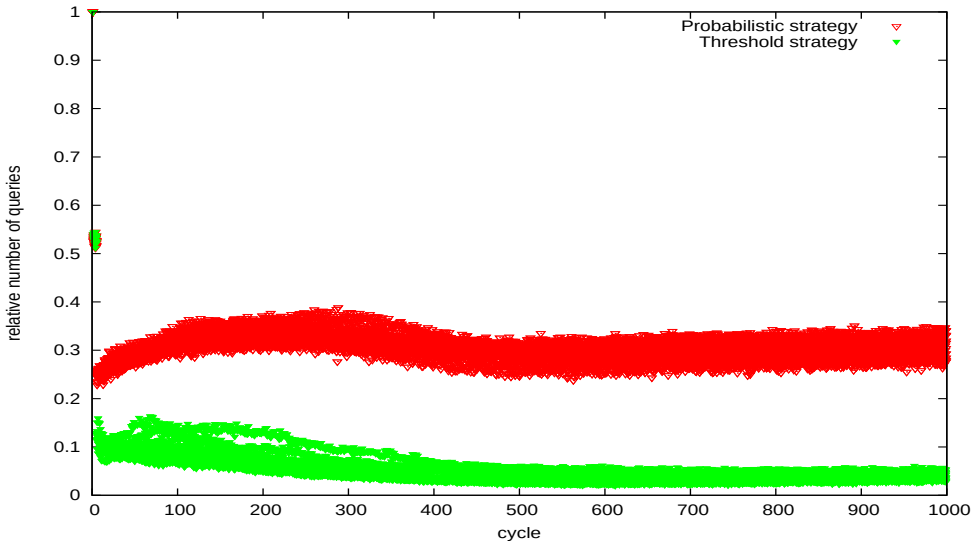


Figure 8.3: Number of queries in % using active strategy 1 (Threshold) and 2 (Probabilistic) executed by the SNG algorithm on the Checker data set.

	SNPC		$\text{SNPC}_{\text{active strategy 1}}$			$\text{SNPC}_{\text{active strategy 2}}$		
	Rec.	Pred.	Rec.	Pred.	Rel. #Q	Rec.	Pred.	Rel. #Q
WDBC	90%	89%	66%	93%	60%	82%	92%	15%
Proteom_2	71%	81%	72%	82%	67%	65%	82%	21%

	SNPC-R		$\text{SNPC-R}_{\text{active strategy 1}}$			$\text{SNPC-R}_{\text{active strategy 2}}$		
	Rec.	Pred.	Rec.	Pred.	Rel. #Q	Rec.	Pred.	Rel. #Q
WDBC	86%	91%	85%	94%	62%	94%	95%	12%
Proteom_2	72%	86%	89%	82%	66%	92%	87%	16%

Table 8.3: Classification accuracies for cancer data sets using standard SNPC and SNPC-R. Data characteristics as above. The prediction accuracies show a reliable good prediction of data belonging to the WDBC data as well as for the Proteom_1 data set. By use of relevance learning the number of queries as well as the prediction accuracy improved slightly with respect to the standard approach.

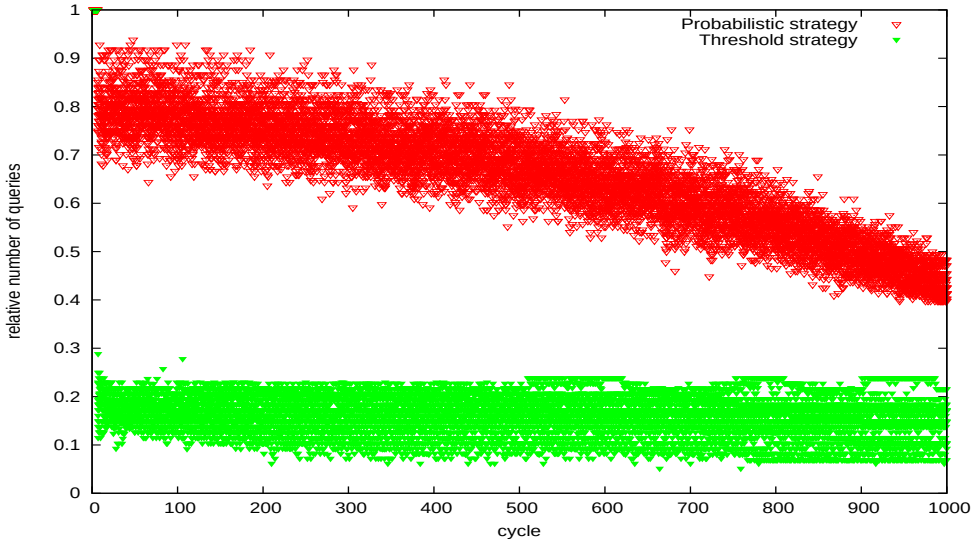


Figure 8.4: Number of queries in % using active strategy 1 (Threshold) and 2 (Probabilistic) executed by the SNPC-R algorithm on the Proteom$_2$ data set.

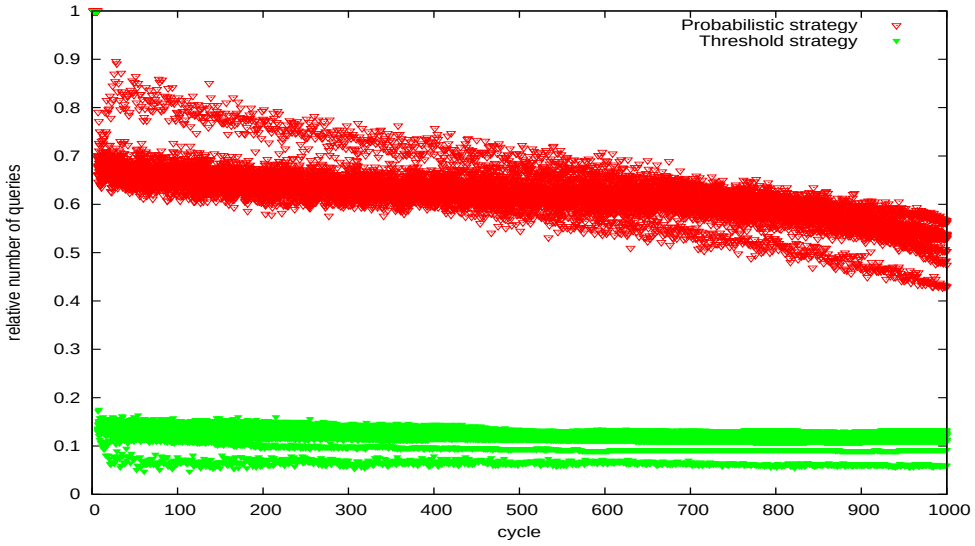


Figure 8.5: Number of queries in % using active strategy 1 (Threshold) and 2 (Probabilistic) executed by the SNPC-R algorithm on the WDBC data set.

For the WDBC data set and the Proteom$_2$ data set we found small improvements in the prediction accuracy using the active strategy 2. In parts small over-fitting behavior using the new query strategies can be observed. Both new query strategies were capable to significantly decrease the necessary number of queries by keeping at least reliable prediction accuracies with respect to a random query approach. This is depicted for different data sets using the SRNG algorithm in Figure 8.1, 8.2 and 8.3 and in Figure 8.5 using the SNPC algorithm. Although the number of queries differs for

SNPC and SRNG on the same data set the generic trend is similar but clearly reflects the classification performance of the individual classifier. Especially the threshold approach let to a significant decrease in the number of queries in each experiment.

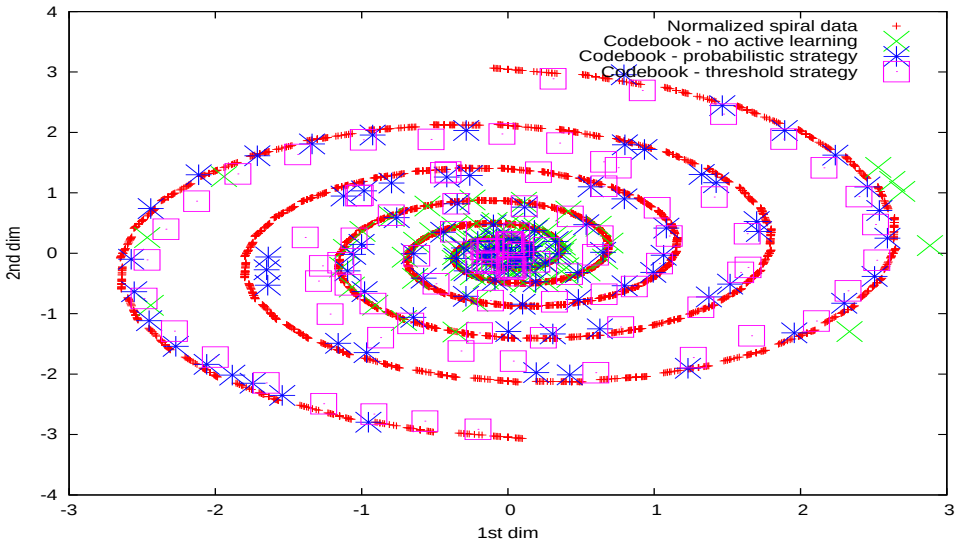


Figure 8.6: Plot of the synthetic spiral data with prototype positions as obtained using the different query strategies.

In Figure 8.6 the behaviour of the proposed strategies are shown for a synthetic spiral data set of around 5000 data points. Both active learning strategies showed a significant decrease in the number of necessary queries, typically only $10\% - 30\%$ of the queries were executed with respect to a random approach. Using the active learning strategies the data could be learned nearly perfect. Random querying required a much larger number of cycles to obtain acceptable results. Considering the prototype distribution one clearly observes the well positioning of the prototypes using the active learning strategies whereas the random approach suffers by over-representing the core of the spiral which has a larger data density.

Both active learning strategies show reliable or partially better results in their generalization ability with respect to the random approach. Thereby a significantly faster convergence with a much lower number of necessary queries can be found. For the *threshold strategy* an overall stable behavior with good prediction rate and a significant decrease in processing time has been obtained. Due to the automatically adapted parameter the strategy is quite simple but depends on a sufficiently well estimation of the local data distribution. By scaling the threshold parameter an application specific choice between prediction accuracy and speed can be obtained. The *probabilistic strategy* has been found to get similar results with respect to the prediction accuracy but the number of queries is quite dependent of the annealing strategy, simulated less restrictive constraints showed a faster convergence but over-fitting on smaller training data sets. Especially, for larger data sets the proposed active learning strategies show

great benefits in speed and prediction as shown e.g. for the WDBC data set. Especially for the considered mass spectrometric cancer data sets an overall well performance improvement has been observed. This is interesting from a practical point of view, since the technical equipment for measuring e.g. a large number of mass spectrometric data becomes more and more available. In mass spectrometry it is easy to measure a sample multiple times. The replicates which are taken from such multiple measurements are in general very similar and do differ only by random but not by systematic new information. In clinical proteomics based on mass spectrometry replicates are measured very often to decrease the measurement variance (e.g. by averaging) or to reduce the loss of missed samples in case of an error during a measurement of a sample. Typically $4, 8$ or even 16 multiple measurements of the same sample are generated and hence also for moderate sample sizes (e.g. 50 samples per class) the amount of training data becomes huge[2] The presented approach is optimal suited to deal with replicate measurements which may drastically increase the number of samples and hence typically lead to very long runtimes for ordinary trainings using the considered classification algorithms.
The proposed growing strategies have been analyzed in an initial simulation using the two spiral problem c.f. Figure (8.7) have been performed. This classification benchmark has been widely used before, so that results for comparison are readily available. The data set consists of 194 points arranged on two interlaced spirals in the plane. Each spiral corresponds to one class. Due to the high nonlinearity of the task it is particularly difficult for networks consisting of global units (e.g. MLP's). However, the varying density of data points (which is higher in the center of the spirals) make it also a challenge for networks of local units [47]. The test set of 579 points consists of three points between each pair of adjacent same-class training points. Let $\mathbf{d}$ be the mean distance vector between an adjacent pair $(\mathbf{p_1}, \mathbf{p_2})$, then each triple is defined as $(\frac{1}{2}(\mathbf{p_1}+\mathbf{p_2}), \mathbf{p_1}+\mathbf{d}, \mathbf{p_1}-\mathbf{d})$ with removed duplicates. In Figure (8.7) a typical network generated by our method can be seen as well as the corresponding Voronoi tessellation with 34 prototypes. The algorithm was able to learn the data characteristic with 98% prediction accuracy with an initial setting of two randomly positioned prototypes.

The algorithm has also been applied to the Checkerboard data. For simplicity this data set was used in a reduced form with a 4×4 grid. The Checkerboard data set is highly multi-modal and is sufficiently represented by at least 16 prototypes. The algorithm has again been started with only 1 prototype per class located in the origin, and the data had been split into training and test data using each 2nd point for the test data set. After convergence (max 2000 cycles) the obtained LVQ network consists of 17 prototypes which leads to a 100% recognition and prediction on the data. The final network is shown in Figure (8.8) and is close to an optimal configuration with only one redundant prototype.

Beside the applications on synthetic data we applied the algorithm on real life data taken from cancer studies. The first real life data set is the *Wisconsin Breast*

[2]Considering 50 samples per class and 16 replicates per sample one would be confronted with 1600 highly redundant training items.

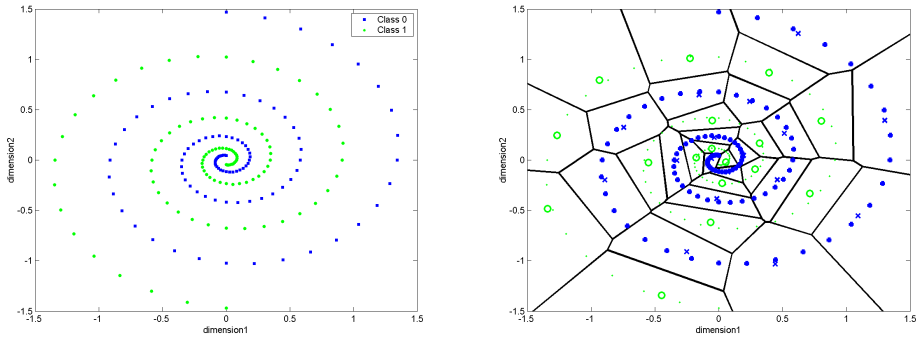


Figure 8.7: Left the original spiral data set [86] is shown. The right plot shows the dynamic adapted SRNG network including decision bounds.

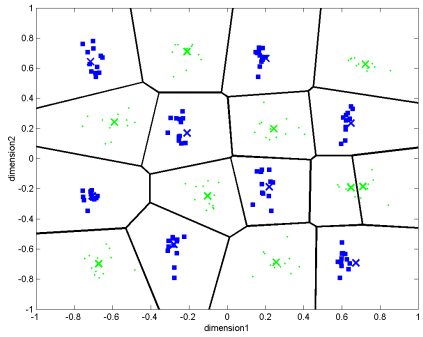


Figure 8.8: Checkerboard data with dynamic adapted SRNG network.

Cancer-Data set (WDBC) as given by UCI [13]. It is processed with 376 samples for training and 197 sample for test. The final networks reduced to the most two separating dimensions determined from the obtained λ relevance vector are shown in Figure (8.9). The WDBC data could be predicted nearly perfect with 97% prediction accuracy using 16 prototypes.

The remaining two data sets are taken from proteomic profiling studies using the Proteom 1 and Proteom 2 data set. Again we started with one prototype per class. The obtained results in comparisons with a SVM obtained using libsvm with a ν-SVM are shown in Table (8.4)[3].

The proteom data can not be sufficiently separated in two dimensions, hence a potential biomarker pattern is more complex. Nevertheless the obtained network for the data set Proteom I & II could successfully model the underlying classification problem with a prediction accuracy of 84% for Proteom I and 92% for Proteom II. For WDBC as well as for the Proteom data the former used settings for the number of prototypes per class based on pure heuristics showed lower performance or were more complex with respect to the number of prototypes by a similar prediction accuracy

[3]For SVM a 10 fold crossvalidation was applied using an optimized polynomial kernel

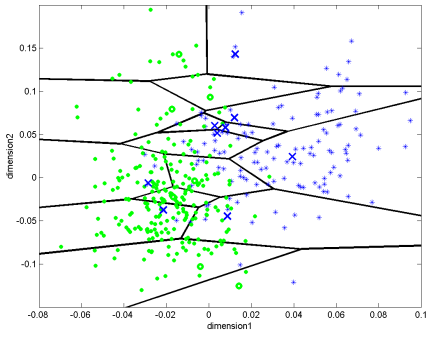


Figure 8.9: The dynamic created SRNG network for the WDBC data set

Data sets	# Prototypes Dyn. SRNG	Pred. Dyn. SRNG	Pred. SVM
Proteom 1	10	84%	76%
Proteom 2	6	92%	85%
WDBC	16	97%	87%

Table 8.4: Prediction accuracies using a SVM classifier and dynamic SRNG for the proteomic data

(e.g. WDBC with 97% and 20 prototypes [172]).

For the UCI and the proteomic data the automatic adapted network structure is especially useful because a visual inspection of the data is complicated and hence a reliable fixed setting of the network was not possible. The obtained networks gave quite good results in prediction with small codebooks, surprisingly also with respect to an optimized SVM. The methods introduced in the former chapters are now applied for further real world problems taken from proteomic studies in a detailed study.

Part IV

Soft Learning in MS Proteomic Research

Chapter 9

Analysis of Mass Spectrometry Data from Clinical Research

However beautiful the strategy, you should occasionally look at the results.
Winston Churchill

In the recent years, mass spectrometry (MS) has been recognized as a *Gold Standard* tool for the identification and analysis of individual proteins in expression proteomics studies. With the increased usage of MS as a standard tool for the life science applications and the advancement of MS instrumentation, sample preparation and bioinformatics, MS technology has entered novel screening and discovery application areas that are beyond the traditional protein identification and characterization applications. The area of clinical diagnostics and predictive medicine are such two examples of these fields. Predictive markers or biomarker for early diagnosis of diseases are of growing importance for the human health-care community.

The goal of using MS in clinical proteomics is to generate protein profiles (mass to charge [m/z] ratio versus signal intensity) from readily available body fluids like serum, saliva and urine to detect changes in protein levels that reflect changes in the disease states. Biomarker patterns are hidden in complex mass spectra and are not always obvious to the human eye. The improved Learning Vector Quantization approaches introduced in this thesis are especially suited to determine these unique biomarker patterns.

The ability to screen an individual's serum for specific biomarkers is an important tool in the early diagnosis of cancer. This screening procedure enables a non-invasive examination of patients who are predisposed to cancer. Most of the already known biomarkers such as PSA for prostate cancer have problems with *sensitivity* or *specificity*. Sensitivity problems arise if the biomarker protein is not present in all patients or if its level is too low to be detected in early stages of the disease. Specificity problems arise due to the expression of the respective protein in tissues other than the cancer of interest. Individual biomarker are in many cases insufficient for a valid cancer diagnosis and the discovering of biomarker patterns may give a more valid diagnosis and

may appear before the onset of symptoms. By the success of mass spectrometry, especially matrix assisted laser desorption/ionization-time-of-flight (MALDI-TOF) MS , one is capable to screen and discover multiple biomarkers simultaneously in clinical proteomic. MALDI-TOF MS has enabled investigators to analyze, in parallel, a large number of serum proteins and peptides from patients. Therein the biomarkers are detected by an intrinsic property of the analyte: its molecular weight. These diagnostic biomarkers are discovered by the use of sophisticated pattern determination and class prediction algorithms.

The ideal biomarker should have a high reproducibility in each population affected by the considered disease and should give high sensitivity and specificity. It should be an indicator for different stages of the disease and should become especially affected by treatment of the disease so that for recovered patients the biomarker is vanishing. This properties are very hard to achieve and depend on a lot of parameters. In our analysis we mostly ignore the specific kind of sample generation and preparation and mainly focus on the Learning Vector Quantization Algorithms as one main part in the processing queue.

Specific improved versions are used for identification of relevant features taken from the protein profile and for classification of new data against a predetermined model. The proposed methods focus on the special problems involved in the biomarker research in clinical proteomic. Most diseases are characterised by different, often overlapping stages which lead to multi class classification tasks with unclear label information. Therefore localized classification models are considered to be more appropriate to deal with clinical data. In the following we apply the proposed improved localized Learning Vector Quantization methods to clinical data showing clear benefits and giving more knowledge about the detailed data distributions. The dynamic labeling approach for Learning Vector Quantization was developed to handle data sets with overlapping distributions giving more information about the safety of a classification.

This is very important for clinical diagnosis. In addition, metric adaptation is used to handle specifics of the underlying data sets and to identify the relevant profiling features which may indicate biomarker patterns.

At the beginning, each new set of data can be considered to be unknown. Especially we do not know the real dimensionality of the problem, the relative position of the (sub)classes in the high-dimensional data space and the appropriate metric. Therefore it is useful to estimate the intrinsic dimension (see Chapter 3) of the data in a first stage before doing more advanced analyses.

The two dimension estimators have been applied upon the proteomic data WDBC and Proteom I and II. Thereby it was observed that the different methods show variations in their estimation. But in general, a common fix point as number of dimension can be found for each data set. The WDBC data set has an estimated intrinsic dimension of 6 with a variance of 1. This shows that although the data set appeared to be not very complex from the former results, it has in fact a potential large dimensionality. Both, the Proteom I and II data sets have an intrinsic dimension of 10, which is interesting because the data are in general 10 times larger in the sense of peak dimensions.

This result is supported also by a PCA, which indicates a similar number of principal components to be important with respect to an explanation of variance. From the obtained estimation results it can be concluded that the proteom data sets under analysis are high-dimensional and subsequent analysis will be complex.

9.1 Self Organizing Maps to visualize high-dimensional data

The characteristic feature distinguishing neural maps from other neural network paradigms and from the regular vector quantization schemes like the LBG-method or its related k-means clustering, is that a topological structure is introduced in the set A. In the context of neural maps the elements of the set A are called neurons. The topological structure allows a weight vector adaptation, which includes neighborhood cooperativeness. Under certain conditions, neural maps create a topology preserving or topographic mapping, which roughly speaking means that the topological relations between data vectors v are preserved during mapping. Several architectures are known: the Elastic Net [36, 179], the Generative Topographic Model (GTM) [12], the Self-Organizing Map (SOM) [83] and the Topology Representing Network (TRN) [95] to mention just a few. Here we will use the SOM. Considering the Proteom I and II data set the application of the SOM algorithm leads to a sufficient planar visualization of the high-dimensional data in a two dimensional grid (c.f. 9.3 and 9.4). By calculating the topographic product [5] of both maps, one could observe, that the maps for the Proteom II data set is topologic preserving and the map for the Proteom I data set is preserving but only on a low level. Hence the map for the Proteom I data set has to be interpreted carefully. Such we can conclude from the visual inspection of the planar map and transfer these to the high-dimensional data set. In figure 9.1 a plot of four component planes for specific feature dimensions is shown. Thereby one observes a good separation of the two classes on the map. The component planes reflect the already known class labeling and can partially be used to identify relevant input dimensions. This however requires manual inspection, which is in general not adequate for a larger number of features. For the component planes of the Proteom I data set (c.f. 9.1) we observe that also the three given class labels are already well separated on the component plane by considering the different intensity levels of e.g. the second feature. However we also observe that one feature alone is not sufficient to give a perfect separation. This becomes also obvious by considering the data together with the map. To do so the SOM has been projected together with the data in the PCA space spanned by the first two principal components. For the Proteom I data set (c.f. 9.4) we find a reliably good separation in the projected space. One can observe that the map is folded quite much. For each data set a good positioning of the prototypes of the SOM has been found.

On these proteom data sets with already known class structure the SOM has been

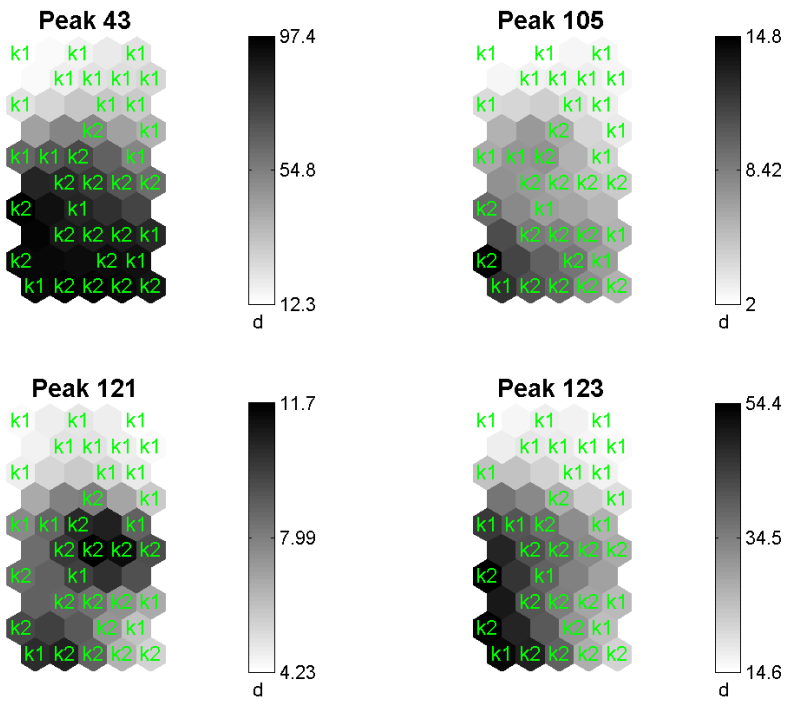


Figure 9.1: Component planes of four relevant input dimensions for the Proteom II data set. The SOM cells are labeled using a majority voting scheme. A reliable good separation of the two classes, considering the different intensity levels and the map topology, can be observed.

found to give perfect visualization, which reflects the underlying classification structure, although this information has not been used during the SOM training. One also observed that the component planes for the individual input dimensions partially give sufficient information to see the class structure on the map. This clearly shows that by explorative analysis for different clinical parameters the component planes can be a reliable good tool, which may help to identify unknown class information on the data.

9.2 Unsupervised analysis on MS -data

In addition to the previously mentioned unsupervised SOM it can be beneficial to apply some clustering analysis upon the given data. For example one could analyse the samples with respect to some non pathologic criteria e.g. the age of the patients. Thereby the number of classes and the specific ranges are unknown.

During the last decades a lot of different clustering algorithms have been proposed ([1] for an overview). All these methods have different pro and cons. As one possible

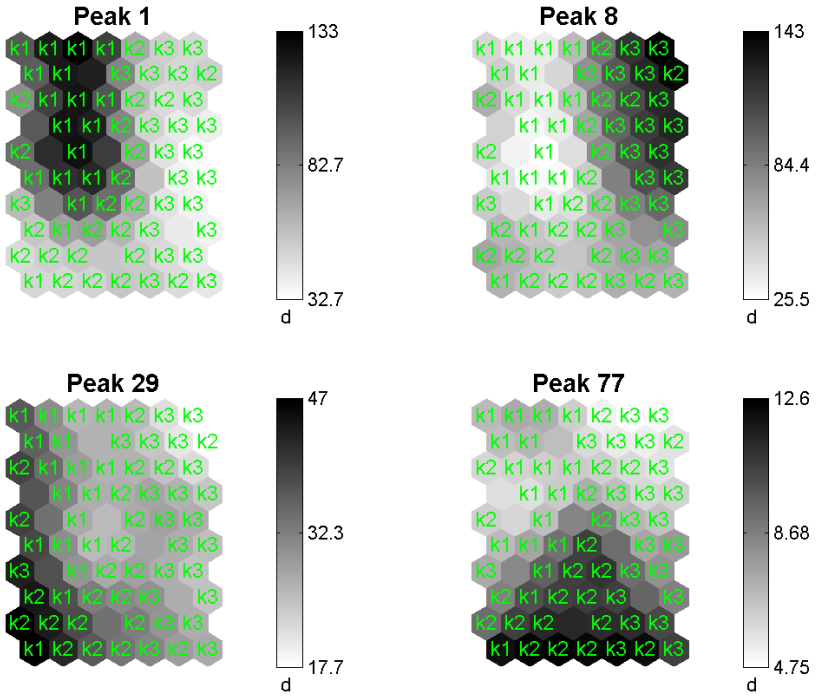


Figure 9.2: Component planes of four relevant input dimensions for the Proteom I data set. The SOM cells are labeled using a majority voting scheme. A reliable good separation of the different classes, considering the different intensity levels and the map topology, can be observed.

clustering algorithm the already introduced and well understood Neural Gas algorithm in a recently proposed very fast batch variant ([31]) can be applied. The clustering can be used to search for clusters in the data in an unsupervised explorativ manner. If this procedure is combined with a subsequently post labeling one may be able to answer some questions about the data as mentioned above. In addition also classical approaches such as a Principal Component Analysis (PCA) can be helpful to analyse the data with respect to an analysis of variance. We tried the approach on some proteomic data were elementary clinical parameters were available but we only found the already expected main class information. The principle procedure is to apply the batch neural gas with a varying number of prototypes and an optional alternative metric such as scaled euclidean or the correlation measure as introduced in [105]. Then the clinical parameters are mapped to the post labeled data points connected to their prototypes. Subsequently the receptive fields on individual prototypes has to be evaluated and some elementary statistic has to be applied to get an impression of the cleanness of prototypes. Hence, one would count the number of common labels for points assigned to a prototype or may use histogram plots if the labels of the points are not discrete

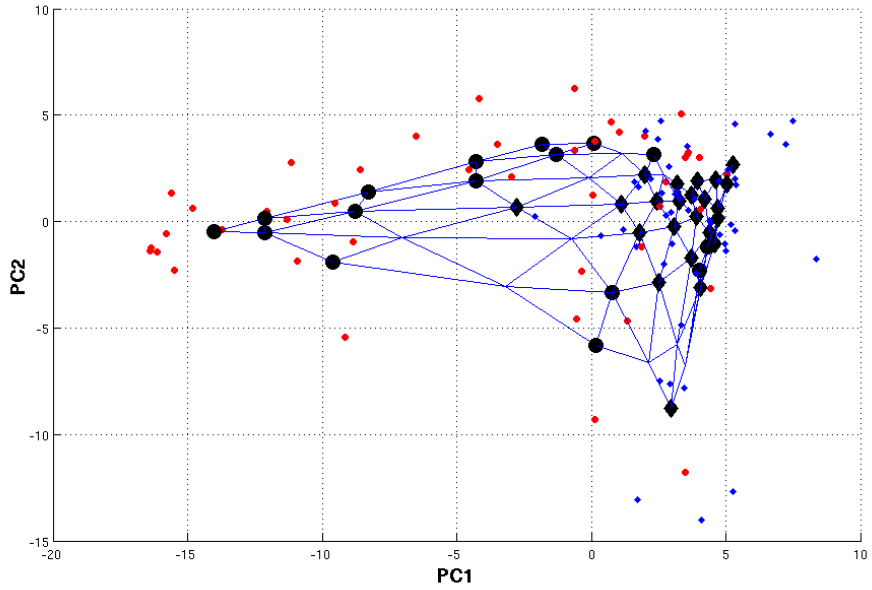


Figure 9.3: Projection of the folded SOM together with the Proteom II data using the first two principal components. One class of the data is plotted with ∘ and the other one with ♢. The prototypes of the SOM are connected by lines and have been labeled using a majority voting scheme. The class label of a SOM prototype is indicated again by ∘ or ♢ but double sized.

such as e.g the body weight or blood pressure measures. This explorative analysis can become quite complicated especially by considering different bead preparations but is sometimes the only way to switch from an unsupervised to a supervised analysis.

9.3 Classification of MS -data

The main purpose of this work is to establish powerful classifiers, which give discriminant models for prediction of e.g. disease stages. The derived methods are dedicated to adapt the metric in accordance to the classification problem and to give a crisp classification result by a nearest winner decision with crisp prototypes or a fuzzy decision by a possibilistic decision using the introduced interpolation scheme in combination with the learned fuzzy labeling of the prototypes. The prototype based models are compared with classification models, which are obtained from classical methods. In detail the results will be compared with support vector models [153] as an alternative margin based soft classifier and the Linear/Quadratic Discriminant Analysis as two statistical approaches. The model evaluation takes place by use of a simple crossvalidation procedure splitting the data into a training $(2/3)$ and a test set $(1/3)$. All data are normalized such that all features are $N(0,1)$ distributed. If not

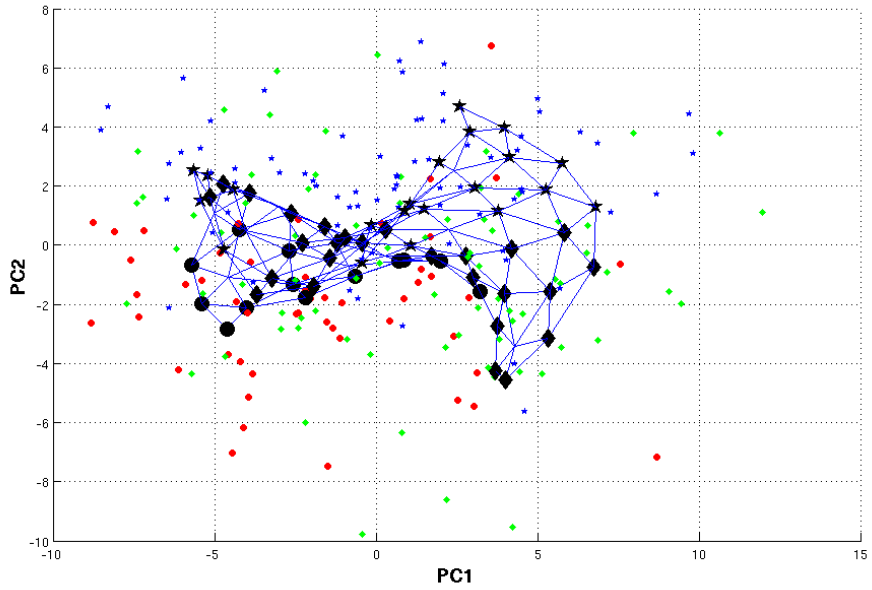


Figure 9.4: Projection of the folded SOM together with the Proteom I data using the first two principal components. One class of the data is plotted with $\circ$, the second using $\Diamond$ and the third using $\star$. The prototypes of the SOM are connected by lines and have been labeled using a majority voting scheme. The class label of a SOM prototype is indicated again by $\circ$, $\Diamond$ or $\star$ but double sized.

stated otherwise the prototype based approaches have been initialized using Batch-NG, the upper number of cycles is limited to 10000, for non-standard SRNG and non-standard SNPC active learning has been used as introduced before. The number of prototypes is specified at the experiments and split equal to the number of classes. Further the following parameter settings were fixed: weight vector learning rate $\epsilon_+ = 0.01$, correct prototype $\epsilon_- = 0.05$ incorrect prototype, metric learning rate $\epsilon_\lambda = 0.01$ and neighborhood scaling $\sigma = N_W/2.0$. All learning rates are exponential annealed during learning.

There are a lot of other well known classification algorithms ([91] for a review) but we will restrict our analysis on the above approaches. Thereby, the main goal of the analysis is to show the specific properties and usefulness of the new possibilities introduced by the improved prototype based methods.

9.3.1 Classification with SRNG and Alternatives

For the Proteom II data set we obtain a prediction accuracy of 81% using LDA and 71% using QDA with the first 30 (lower masses) features. The corresponding confusion matrices are shown in Figure (9.5). For the Proteom I data set only an LDA has

$$\begin{pmatrix} 1.0 & 0.0 & 0.0 \\ 0.1 & 0.74 & 0.16 \\ 0.0 & 0.1 & 0.8 \end{pmatrix} \quad \begin{pmatrix} 1.0 & 0.1 & 0.2 \\ 0.1 & 0.7 & 0.3 \\ 0.0 & 0.3 & 0.7 \end{pmatrix} \quad \begin{pmatrix} 0.80 & 0.2 \\ 0.18 & 0.82 \end{pmatrix} \quad \begin{pmatrix} 0.80 & 0.2 \\ 0.33 & 0.67 \end{pmatrix}$$

Figure 9.5: Confusion matrix for Proteom I using LDA matrix 1, QDA matrix 2 and on Proteom II with LDA matrix 3 and QDA matrix 4.

Data sets	# Support vectors	Pred. SVM	# Prototypes	Pred. SRNG
Proteom 1	203	85%	9	78%
Proteom 2	26	89%	6	89%

Table 9.1: Prediction accuracies using a standard SVM classifier on the proteomic data in comparison to results as obtained using standard SRNG with scaled Euclidean metric. All results are calculated on the full number of features.

been calculated upon the first 30 features. This restriction was necessary to make LDA/QDA applicable, because for a larger number of features the method became numerically instable. We obtained a prediction accuracy of 81% and with QDA of 75%. The corresponding confusion matrices are shown in Figure (9.5). With SVM using a standard SVM and a polynomial kernel, we obtained 82% prediction accuracy for Proteom I and 89% prediction accuracy for Proteom II. The results are summarized in Table (9.1):

In addition to an analysis with standard approaches and the original SRNG, we applied the formerly introduced CFE method upon the proteom data. For the Proteom II data set we found small improvements by use of the CFE method in comparison to usage of the ordinary ranking as obtained by SRNG. The prediction accuracy determined by crossvalidation are depicted in Figure (9.6) (left) and the individual results for SRNG and SRNG+CFE are shown in Figure (9.3.1).

The benefit of a SRNG with CFE application get stronger support for the Proteom I data set were the CFE approach is almost all better than the ordinary SRNG ranking considering the interesting feature set sizes $1 - 25$ (c.f. Figure (9.6), right plot). The individual results for Proteom I are shown in Figure (9.8).

Taking these results into account, the SRNG evolved by Tanimoto similarity measure and the Mahalanobis metric has been applied to the Proteom I and Proteom II data set. Thereby for the Mahalanobis metric it could be observed - in perfect match to initial results given in [172] - that the Mahalanobis metric is strongly effected by a large number of noise dimensions. This however is a typical fact in the analysis of proteom spectra where a large number of features is neglectable and can be considered as noise with respect to the classification task. Using SRNG with Mahalanobis metric on the whole set of features, the algorithm performed very badly and instable. But after a selection of the best four features in accordance to the CFE result, the performance

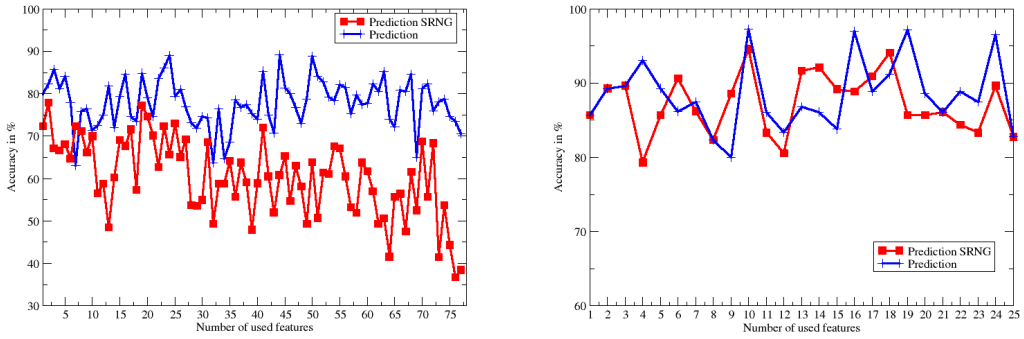


Figure 9.6: Prediction accuracy of the Proteom I (left) and Proteom II (right) data using SRNG and SRNG+CFE with scaled Euclidean metric. The number of features is varied in accordance to the corresponding ranking.

Data sets	Pred. SRNG+TM	Pred. SRNG+MM	Pred. SNG	Pred. SRNG
Proteom I	78%	78%	78%	78%
Proteom II	65%	90%	74%	89%

Table 9.2: Prediction accuracies using Tanimoto (TM) and Mahalanobis (MM) metric in comparison to ordinary S(R)NG with scaled Euclidean metric on the proteomic data

significantly improved. The results are given in Table (9.2).

9.3.2 Localized Classifiers for Multidimensional Distributions

Now the specific results for local metric adaptation of the different derived methods are analyzed. Thereby we consider the scaled Euclidean metric in a class wise local sense such that one obtains three metrics for the Proteom II data set and two metrics for the Proteom I data sets for the respective classes. The results for the different classifier types with metric adaptation are depicted in Table (9.3). The number of prototypes has been determined in accordance to the former obtained results from the growing learning studies upon the data.

The results show improvements in the prediction accuracy with respect to non-metric adaptive NPC algorithms and also with respect to SRNG. Local metric adaptation did not get additional improvements in the prediction accuracy but are useful to identify class specific attributes in the peak set. Using LSNPC local relevance profiles can be obtained. Local relevance profiles for the Proteom II data sets are depicted in figures 9.9 and 9.10. The local relevance profiles can be compared with respect to a global relevance profile as shown in figure 9.11. Some of the important peaks already identified in the unsupervised analysis (c.f. 9.1) can be found again in the global rele-

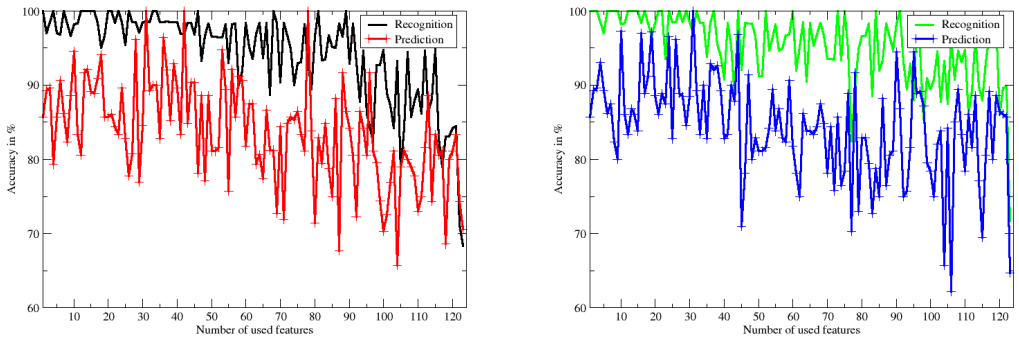


Figure 9.7: Prediction vs. Recognition of the Proteom II data using SRNG and SRNG+CFE with scaled Euclidean metric. The number of features is varied in accordance to the corresponding ranking.

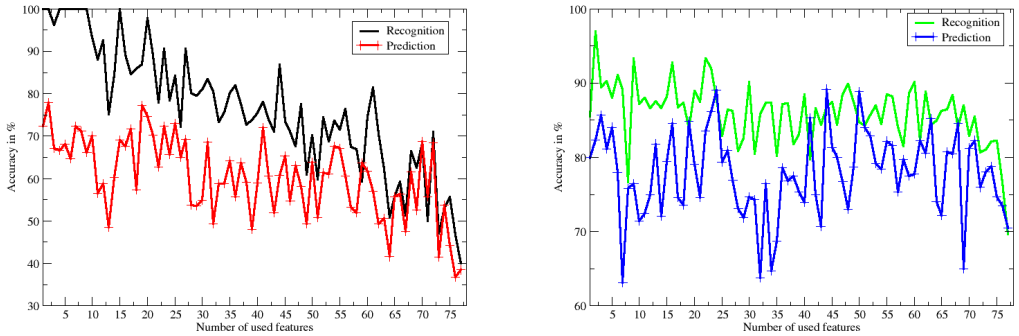


Figure 9.8: Prediction vs. Recognition of the Proteom I data using SRNG and SRNG+CFE with scaled Euclidean metric.

vance profile (e.g. peak 105 and peak 43). Interestingly these peaks are also indicated by high relevance values in the local relevance profile (peak 105 and 121 in figure 9.10) but in a disjunct manner. Both local profiles are quite dissimilar. So the local relevance profile may give additional hints to which class an important peak mainly belongs. This can be related to the post-hoc test procedures from classical statistics, which can be applied for multi class analysis to identify the source of variance of a considered feature. However the relevance profiles should not be over interpreted, because the relevance profiles are in generally not completely stable and may vary slightly between different runs on the same data set. Also it should be kept in mind that the weighted Euclidean metric used herein considers each dimension independently but the peaks of the spectra may be correlated because the underlying peptide expressions could be dependent.

Data sets	Pred. LSRNG	Pred. LSNPC	Pred. SNPC-R	Pred. SRNG
Proteom I	76%	87%	87%	78%
Proteom II	85%	91%	93%	89%

Table 9.3: Prediction accuracies using localized Euclidean metric for SRNG and SNPC in comparison to standard SRNG and SNPC with global relevance learning on the proteomic data

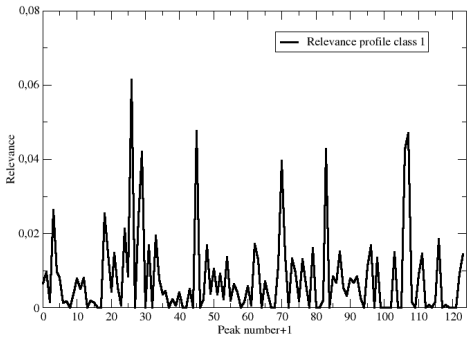


Figure 9.9: Relevance profile of class 1 for the Proteom II data set by use of the LSNPC algorithm.

9.3.3 Supervised Learning with fuzzy Label Information

Subsequently FSNPC derivates are considered with and without metric adaptation for the different data sets. As a first result from the simulations one can found that the fuzzy variants in general need longer runtimes up to convergence, especially to sufficiently learn the underlying labeling. This can be explained due to the label learning of the prototypes, which not any longer is fixed from the startup such that the number of prototypes dedicated to represent a class can be determined during learning. The results depicted in Tab. 9.4 show reliable but a bit worse results with respect to the non fuzzy methods. The local metric adaptation within FSNPC did not improve the prediction accuracy further, but nevertheless the results are reliable good.

Data sets	Pred. FSNPC	Pred. LFSNPC	Pred. FSNPC-R
Proteom I	70%	87%	87%
Proteom II	92%	91%	93%

Table 9.4: Prediction accuracies using localized or global relevance learning with Euclidean metric for F-SNPC on the proteomic data

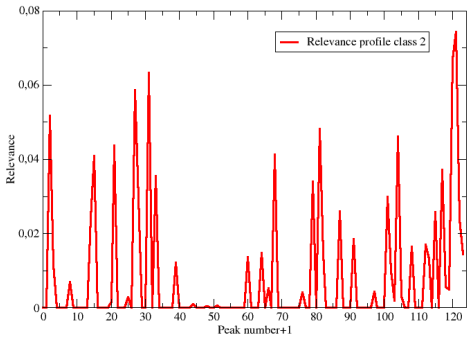


Figure 9.10: Relevance profile of class 2 for the Proteom II data set by use of the LSNPC algorithm.

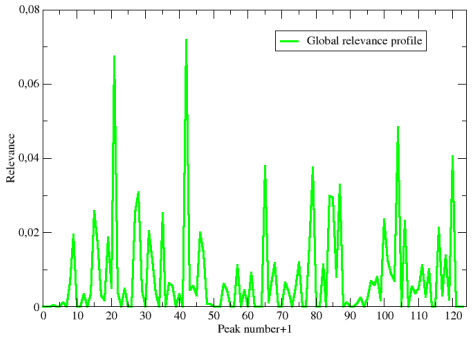
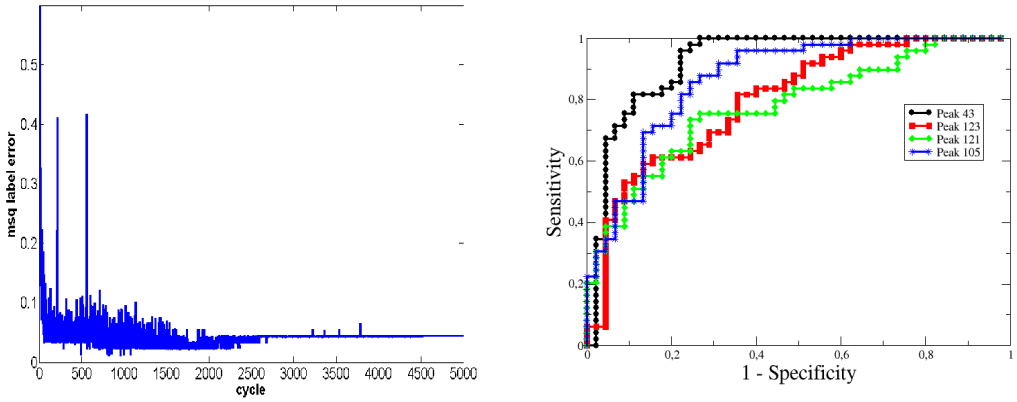


Figure 9.11: Global Relevance profile of the Proteom II data set by use of the SNPC-R algorithm.

In the analysis of the fuzzy method one observes an improvement of the recognition and prediction accuracy by incorporating metric adaptation as depicted in Tab. 9.4. For the fuzzy methods an additional measurement of convergence and accuracy, the mean square label error (LMSQE) becomes important. If the data could be sufficiently well represented by the prototype model the LMSQE is a comparable measure for different models originating from prototype fuzzy classifiers. An initial result is depicted in Figure 9.12(a) giving a first impression of LMSQE behavior for the FLSNPC-R algorithm on the WDBC data set. The LMSQE in combination with the classification accuracy can be used as an indicator for the raw number of prototypes, which should be used to get a sufficient modeling of the underlying data labeling and by considering this measure over time is a less raw measure for the current algorithm convergence than the pure accuracy, which typically is constant over large periods of learning. The algorithm shows an overall convergence of the LMSQE and ends up with a small error

value. However for the FSNPC-R one finds a less stable behavior reflected by strong fluctuations in the middle of the learning task, which are vanishing in the convergence phase. One can also observe that the FSNPC-R algorithms gets a low LMSQE already at a very early cycle. Considering the fuzzy labeling of the final prototype sets one can see that the algorithm was capable to learn the labeling from the given training data. One finds prototypes with a very clear labeling, close to 100% for the corresponding class and hence a quite clear voronoi tessellation induced by this prototypes. But one can also find prototypes with lower safety in its class modeling and even prototypes, which show split decisions. Especially the last ones are interesting in the sense that one immediately knows that decisions taken by those prototypes are doubtful and should be questioned.



(a) LMSQE (b) ROC

Figure 9.12: Typical convergence curve for mean square label error (LMSQE) using FSNPC-R for the WDBC data. To get a more stable analysis the algorithms have been trained fix with 5000 cycles to obtain these LMSQE curves using 6 prototypes. The second figure shows a ROC analysis for the most important four peaks found in the Proteom II data set. The curves are generated using the corresponding peak areas as a raw criterion for classification

The different methods have in common that relevance learning is supported such that classification specific important input dimensions, or in our case peaks, can be easily identified for a subsequent analysis. If a two class scenario is available such as the Proteom II data set one is able to screen out the potential relevant input dimensions in more detail under clinical constraints. As already explained one would prefer peaks as potential biomarkers, which show high sensitivity and specificity in common. In Figure 9.12(b) the results of a ROC analysis for the already identified peaks (peak $43, 105, 121, 123$) are shown. The four peaks clearly show a different behavior under the ROC analysis making them useful for clinical analysis. The common

Classifier	SNG	SNG	SNG	GLVQ	GLVQ	SNPC	SNPC	SNPC	FSNPC	FSNPC	FSNPC	QDA	LDA	SVM
Metric	Euc	Euc_λ	Euc_{λ_r}	TM	MM	Euc	Euc_λ	Euc_{λ_r}	Euc	Euc_λ	Euc_{λ_r}	MM	MM	Poly
WDBC	95	97	96	96	89	89	95	97	93	91	95	94	96	87
Proteom_1	78	84	76	78	78	79	87	87	70	87	87	75	81	76
Proteom_2	87	93	85	65	90	82	87	91	92	93	91	71	81	85

Table 9.5: Collected prediction accuracies in $\%$ for the different clinical data sets using the introduced algorithms. The used metrics are Euclidean (Euc), scaled/parametrized Euclidean (Euc_λ), local scaled Euclidean (Euc_{λ_r}), Mahalanobis metric (MM), Tanimoto metric (TM) and a polynomial kernel (Poly) with $\gamma = \frac{1}{1000}, r = 1.0, d = 2.0$. The results are calculated using the simple or 10-fold crossvalidation. Parameter settings as already defined. Active Learning and initialization with Batch-NG has been used for classifiers with scaled Euclidean metric. For the classification models with Tanimoto or Mahalanobis metric the proteomic data are used with the best 4 peaks (defined by SRNG+CFE) only.

measure AUC (area under the curve) makes clear that peak 42 is most beneficial with a quite high sensitivity and specificity. Clearly, the ROC analysis can be used to screen out identified relevant peaks under a more restrictive clinical univariate view. This is typically a subsequent step to derive a clinical applicable solution from the identified marker candidates by use of less complex but certified standard classification approaches.

In Table 9.5 multiple results for the different classifiers and data sets are collected in a common view. Nearly all classifiers show good results of $> 90\%$ for the WDBC data set. The Proteom_1 data set is the most complicated data set and is modeled best using a classifier with relevance learning. The Proteom_2 data set is in general well modeled by the different classifiers but also in this case the usage of relevance learning typically improves the prediction accuracy. If classifiers with fuzzy labelings are used the prediction results remain reliable good with respect to non-fuzzy approaches with the additional benefit of easier modeling of the network structure. Localized metric adaptation gives only slight improvements in model accuracy considering SNPC algorithms.

Finally the analysis shows that the performed extensions of prototype based methods could be sucessully applied on real life clinical proteomic data. Thereby the usage of metric adaption is beneficial and can be used to select potential biomarker candidates as well as to improve the prediction accuracy of the classification models. Local relevance learning does not improve the results in general but allows a more specific interpretation of relevant peaks. The fuzzified SNPC allows to get a safety decision of the classification. In addition an improvement of the prediction accuracy could be observed. This is due to a further flexibility in modeling the classifier, because the labels of the prototypes are not any longer fixed and hence only the *number* of prototypes has to be specified. The Tanimoto and Mahalanobis metric are not applicable for

proteomic data using all features but only by a knowledge driven pre-selection. This situation is similar to LDA and QDA, which were only useable with a subset of the features. If the dimensionality is sufficiently reduced reliable classification models could be observed for these types of classifiers. In comparison to the well known SVM the prototype based methods show comparable good or sometimes better results but with the additional benefit of better interpretability.

Chapter 10

Conclusions

Learn the art of patience. Apply discipline to your thoughts when they become anxious over the outcome of a goal. Impatience breeds anxiety, fear, discouragement and failure. Patience creates confidence, decisiveness and a rational outlook, which eventually leads to success.
Brian Adams

10.1 Summary

Localized metrics and fuzzy classification have been studied with self-organizing neural models for supervised processing of synthetic and clinical data originating from mass spectrometry measurements. Different types of Learning Vector Quantizers methods have served as starting points for the extensions to local metrics and fuzzy prototype classifiers presented in this work.

Basically, local metric adaptation has been formally introduced as valid extension of the underlying cost functions and corresponding formal generalization bounds have been derived. The method has been applied under different conditions and the benefits have been shown for synthetic and real life data. Thereby the local metric adaptation shows similar beneficial effects as e.g. for the local estimation of covariances in the quadratic discriminant analysis. The local relevance profiles allow for easy interpretation of relevant input dimensions for individual classes. This is especially interesting for clinical multi class problems were class specific relevant input dimensions may be interpreted as disease specific characteristic separating e.g. a specific cancer from other cancer or control classes.

To make prototype based methods more easily applicable to clinical problems automatic parameter estimations have been developed, which allow for dynamic modelling of the network structure during training. Hence, a network setup can be started on the most simple structure without or very low additional knowledge about the data. This makes the introduced methods much easier to use in clinical proteomics were only minor pre-knowledge about the data may be available.

The capability of long-time learning available with prototype based networks has inspired the search for active learning strategies. Based on the obtained generalization bounds active learning strategies have been derived, which allow for significant faster network training especially if the number of sample increases over time as it can be expected e.g. within long period clinical studies.

Another important issue refers to the safety of classification decisions with respect to confidence and fuzziness. The SNPC algorithm has been adapted to support not only (local) metric adaptation but also to allow fuzzy labeled prototypes. Thereby the labeling of prototypes is adapted during training and gives, combined with an interpolated Voronoi tessellation, safety decisions within a classification. Due to the nature of clinical data a crisp classification is typically only a simplified view on the problem. Multiple clinical indicators lead to fuzziness, which should be modeled appropriate. Using FSNPC this is taken into account by allowing the prototypes to be fuzzy. The classifier reports the clinician a fuzzy decision, which allows for interpretation of the decisions safety.

10.2 Open problems and future issues

Despite the very good classification results and new features supporting a better application of prototype methods for clinical problems some questions and problems remain.

Interpretation and Evaluation As observed in the results metric adaptation is very beneficial to improve the classification models, also the new extensions of LVQ to support Tanimoto or Mahalanobis metric are useful under some constraints but they are currently not directly applicable on the high dimensional data. Using more advanced preprocessing techniques like Wavelet-Analysis gives hope that an adequate compressed representation of the spectral data will be possible, making the direct application of Tanimoto or Mahalanobis metric possible. Further initial results suggest that relevance learning can be further extended by a neighborhood concept such that very MS specific attributes like peak-width can be integrated. This could also lead to improved relevance profiles.

Integrative extensions The combination of CFE and the random forest method [17] could be interesting to optimize not only the ranking but also the classification results. As shown, the local relevance learning has been effectively used to obtain a better modeling of the data characteristics. However, there is still a lack of an appropriate visualization or interpretation method for local relevance profiles. In this context it could be beneficial to improve the BB-tree method as introduced in [54], making use of a global relevance profile to generate an interpretable tree based classifier, by local metrics.

Crossing methodologies A recent approach for prototype based networks supporting the fuzzy labeling [158, 19] of data points could be interesting in the clinical domain as well. Hence, it is interesting, whether the ideas given in [158] can be adapted for FSNPC too. This could also allow to make similar use of a fuzzy labeled SOM as recently introduced in [126], which potentially gives a suitable visualization of fuzzy labeled data or fuzzy prototype networks as mentioned before.

Application fields Beside the typical clinical problem of biomarker detection, alternative clinical problems are also interesting. Such problems (e.g. therapy progress) gain increasing importance and only very few alternative methods exist dealing with such problems. The prediction of a responder / non-responder patient over time is very important, especially for complex cancer types such as Leukaemia. Hence, it could be interesting whether a combination of the methods given in [158, 126] for fuzzy labeling and for handling of sequence data [104] leads to a algorithmic solution for this type of clinical data. Clinical proteomics is analyzing data on the protein level, recent work is focusing on peptides. Hence, it could be interesting whether the introduced methods can be applied in the context of Metabolomics too. This opens a new field of research with even more, high-dimensional complex data.

Appendix A

Derivations

Subsequently the full derivation of the SNPC window rule under the extended cost function for F(L)SNPC is given. This window rule is in analogy to LVQ2.1 and SNPC. For this purpose we consider in (6.7) the term $u_\tau\left(\mathbf{r}|\mathbf{v}_k\right)\left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}} - lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)\right)$ paying attention to the fact that now the $\alpha_{\mathbf{r},c_{\mathbf{v}_k}}$ are fuzzy. Let $\mathcal{LC} = lc\left(\left(\mathbf{v}_k, c_{\mathbf{v}_k}\right), \mathbf{W}\right)$. The term has to be rewritten as $T = u_\tau\left(\mathbf{r}|\mathbf{v}_k\right)\left(\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) - \mathcal{LC}\right)$ and one obtaines

$$
\begin{aligned}
T &= \frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_\mathbf{r})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}\left(\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) - \mathcal{LC}\right) \\
&= \frac{\sum_{\mathbf{r}'}\left(\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) - \alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\left(\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) - \alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)} \cdot \\
&\quad \frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_\mathbf{r})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}} - \mathcal{LC}\right) \\
&= \frac{\sum_{\mathbf{r}'}\left(\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) - \alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)} \cdot \\
&\quad \frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_\mathbf{r})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\left(\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) - \alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}\left(1 - \alpha_{\mathbf{r},c_{\mathbf{v}_k}} - \mathcal{LC}\right)
\end{aligned}
$$

this derivation can be calculated further to a more compact formulation as follows:

$$
\begin{aligned}
T &= \sum_{\mathbf{r}'}\left(\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right)-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}{\sum_{\mathbf{r}''}\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}''})}{2\tau^2}\right)}\cdot \\
&\quad \frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\left(\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right)-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\mathcal{LC}\right) \\
&= \sum_{\mathbf{r}'}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)u_\tau\left(\mathbf{r}'|\mathbf{v}_k\right)\cdot \\
&\quad \frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\mathcal{LC}\right) \\
&= \left(\sum_{\mathbf{r}'}\left(1-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)u_\tau\left(\mathbf{r}'|\mathbf{v}_k\right)-\sum_{\mathbf{r}'}\alpha_{\mathbf{r},c_{\mathbf{v}_k}}u_\tau\left(\mathbf{r}'|\mathbf{v}_k\right)\right)\cdot \\
&\quad \frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}})}{2\tau^2}\right)\cdot\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\mathcal{LC}\right)}{\sum_{\mathbf{r}'}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)} \\
&= \left(lc-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\underbrace{\sum_{\mathbf{r}'}u_\tau\left(\mathbf{r}'|\mathbf{v}_k\right)}_{=1\text{ da W.}}\right)\frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}})}{2\tau^2}\right)\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\mathcal{LC}\right)}{\sum_{\mathbf{r}'}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)} \\
&= \left(lc-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right)\frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}})}{2\tau^2}\right)\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\mathcal{LC}\right)}{\sum_{\mathbf{r}'}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)} \\
&= \left(lc-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right)\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\mathcal{LC}\right)\underbrace{\frac{\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}})}{2\tau^2}\right)}{\sum_{\mathbf{r}'}\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-\alpha_{\mathbf{r}',c_{\mathbf{v}_k}}\right)\exp\left(-\frac{d^\lambda(\mathbf{v}_k,\mathbf{w}_{\mathbf{r}'})}{2\tau^2}\right)}}_{=\Pi\left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right)} \\
&= \left(lc-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right)\left(1-\alpha_{\mathbf{r},c_{\mathbf{v}_k}}-lc\right)\Pi\left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) \\
&= \left(lc(1-lc)-\alpha_{r,c_{v_k}}+\alpha^2_{r,c_{v_k}}\right)\Pi\left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) \\
&= \left(lc(1-lc)-\alpha_{r,c_{v_k}}\left(1+\alpha_{r,c_{v_k}}\right)\right)\Pi\left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right) \\
&= F\cdot\Pi\left(\alpha_{\mathbf{r},c_{\mathbf{v}_k}}\right)
\end{aligned}
$$

with $F=\left(lc(1-lc)-\alpha_{r,c_{v_k}}\left(1+\alpha_{r,c_{v_k}}\right)\right)$.

Obviously, $0\leq lc(1-lc)\leq 0.25$ for lc because of the loss boundary property (6.5), and, hence, $-2\leq F\leq 0.25$ using $\alpha_{r,c_{v_k}}\leq 1$. Now, a similar argumentation as in [135] can be applied: The absolute value of the factor F has to be signifcantly

different from zero to have a valuable contribute in the update rule, which yields the *window condition* $0 \ll |F|$, which can be obtained by balancing the local loss lc and the value of the assignment variable $\alpha_{r,c_{v_k}}$.

Bibliography

[1] M. N. Murty A. K. Jain and P. J. Flynn. Data clustering: a review. *ACM Compututing Surveys*, 31:264 – 323, september 1999.

[2] J. S. Baras and S. Dey. Combined compression and classification with learning vector quantization. *IEEE Transactions on Information Theory*, 45:1911–20, 1999.

[3] P. L. Bartlett, S. Boucheron, and G. Lugosi. Model selection and error estimation. *Machine Learning*, 48:85–113, 2002.

[4] P. L. Bartlett and S. Mendelson. Rademacher and Gaussian Complexities: Risk Bounds and Structural Results. *J. of Mach. Learn. Res.*, 3:463–482, 2002.

[5] Hans-Ulrich Bauer, Klaus Pawelzik, and Theo Geisel. A topographic product for the optimization of self-organizing feature maps. In John E. Moody, Stephen J. Hanson, and Richard P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 1141–1147. Morgan Kaufmann, San Mateo, CA, 1992.

[6] Hans-Ulrich Bauer and Klaus R. Pawelzik. Quantifying the neighborhood preservation of Self-Organizing Feature Maps. *IEEE Trans. on Neural Networks*, 3(4):570–579, 1992.

[7] E.B. Baum. Neural net algorithms that learn in polynomial time from examples and queries. *IEEE Transactions on Neural Networks*, 2:5–19, 1991.

[8] S. Baumann, U. Ceglarek, G.M. Fiedler, and J. Lembcke et al. Standardized approach to proteomic profiling of human serum based magnetic bead separation and matrix-assisted laser esorption/ionization time-of flight mass spectrometry. *Clinical Chemistry*, 51:973–980, 2005.

[9] J. Baxter. The canonical metric for vector quantization. NeuroCOLT NC-TR-95-047, Royal Holloway, University of London, UK, 1995.

[10] L. M. Belue, K. W. Bauer Jr., and D. W. Ruck. Selecting optimal experiments for multiple output multilayer perceptrons. *Neural Computation*, 9:161–183, 1997.

[11] James C. Bezdek and Nikhil R. Pal. Two soft relatives of learning vector quantization. *Neural Networks*, 8(5):729–743, 1995.

[12] Christopher M. Bishop, Markus Svensén, and Christopher K. I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10:215–234, 1998.

[13] C. Blake and C. Merz. UCI repository of machine learning databases., (last visit 01.11.2006). available at: http://www.ics.uci.edu/ mlearn/MLRepository.html.

[14] J.K. Boelke, M. Gerhard, F.-M. Schleif, J. Decker, M. Kuhn, T. Elssner, W. Pusch, and M. Kostrzewa. *ClinProTools 2.0 User Documentation*, 2005. Available in the ClinProt - ClinProTools 2.0 Software package.

[15] T. Bojer, B. Hammer, D. Schunk, and Tluk von Toschanowitz K. Relevance determination in learning vector quantization. In *9th European Symposium on Artificial Neural Networks. ESANN'2001. Proceedings. D-Facto, Evere, Belgium*, pages 271–6, 2001.

[16] A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159, 1997.

[17] L. Breiman. Random forests. *Machine Learning*, 45:5–23, 2001.

[18] J. Bruske and G. Sommer. Topology representing networks for intrinsic dimensionality estimation. In *Proc. ICANN'97, 7th International Conference on Artificial Neural Networks*, volume 1327 of *Lecture Notes in Computer Science*, pages 595–600. Springer, Berlin, 1997.

[19] C. Brüß, F. Bollenbeck, F.-M. Schleif, W. Weschke, T. Villmann, and U. Seiffert. Fuzzy image segmentation with fuzzy labelled neural gas. In *Proc. of ESANN 2006*, pages 563–569, 2006.

[20] F. Camastra and A.M. Colla. Data dimensionality estimation methods: a survey. *Pattern Recognition*, 36:2945–2954, 2003.

[21] F. Camastra and A. Vinciarelli. Cursive character recognition by learning vector quantization. *Pattern Recognition Letters*, 22(6–7):625–629, May 2001.

[22] F. Camastra and A. Vinciarelli. Intrinsic dimension estimation od data: an approach based Grassberger-Procaccia's algorithm. *Neural Processing Letters*, 14:27–34, 2001.

[23] C. Cambell, N. Cristianini, and A. Smola. Query learning with large margin classifiers. In *International Conference in Machine Learning*, pages 111–118, 2000.

[24] G.A. Carpenter and S. Grossberg. Adaptive resonance theory. In M.A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks 2nd Ed.*, pages 87–90. MIT Press, 2003.

[25] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Worst-case analysis of selective sampling for linear-threshold algorithms. In *NIPS*. 2004.

[26] V. Cherkassky, D. Gehring, and F. Mulier. Comparison of adaptive methods for function estimation from samples. *IEEE Transactions on Neural Networks*, 7:969–984, 1996.

[27] J.C. Claussen and Th. Villmann. Magnification control in winner relaxing neural gas. *Neurocomputing*, 63(1):125–137, 2005.

[28] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. In *Advances in Neural Information Processing Systems 1995*, pages 705–712, 1995.

[29] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[30] K. Cottingham. Clinical proteomics: Are We There Yet. *Anal. Chem.*, 75(21):472–476, 2003.

[31] M. Cottrell, B. Hammer, A. Hasenfuß, and T. Villmann. Batch neural gas. In M. Cottrell, editor, *International Workshop on Self-Organizing Maps (WSOM 2005)*, pages 275–282, 2005.

[32] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[33] K. Crammer, R. Gilad-Bachrach, A.Navot, and A.Tishby. Margin analysis of the lvq algorithm. In *Proc. NIPS 2002*, http://www-2.cs.cmu.edu/Groups/NIPS/NIPS2002/NIPS2002preproceedings/index.html, 2002.

[34] S. Dasgupta. Analysis of a greedy active learning strategy. In *NIPS*. 2004.

[35] G. Van de Wouwer, P. Scheunders, D. Van Dyck, F. Wuyts, and P. Van de Heyning. Voice classification by wavelet transforms and fuzzy interpreted lvq networks. In *Proceedings International Symposium on Soft Computing and Intelligent Industrial Automation, Reading, UK, March 26-28*, pages 171–177, 1996.

[36] C. Von der Malsburg and D. J. Willshaw. How to label nerve cells so that they can interconnect in an ordered fashion. *Proc. Nat. Acad. Sci. USA*, 74:5176–5178, 1973.

[37] M. Diaz, K. Bredies, J. Decker, F.-M. Schleif, U. Claus, M. Kuhn, P. Maass, H. Thiele, and F. Laukien. Wavelet transformation applied to the analysis of clinical proteomics mass spectra. In *52st ASMS Conference (ASMS) 2004*, 2004.

[38] E. R. Dougherty. The fundamental role of pattern recognition for gene-expression/microarray data in bioinformatics. *Pattern Recognition*, 38:2226–2228, 2005.

[39] R. O. Duda and P. E. Hart. *Pattern classification and scene analysis*. John Wiley & Sons, New York, 1973.

[40] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[41] S. Dudoit, J. P. Shaffer, and J. C. Boldrick. Multiple hypothesis testing in microarray experiments. *Statistical Science*, 18(1):71–103, 2003.

[42] Adam BL et al. Serum protein finger printing coupled with a pattern-matching algorithm distinguishes prostate cancer from benign prostate hyperplasia and healthy men. *Cancer Research*, 62(13):3609–3614, July 2002.

[43] R.M. Everson and J. E. Fieldsend. Multi-class roc analysis from a multi-objective optimisation perspective. *Pattern Recognition Letters*, 27(8):918–927, 2006.

[44] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[45] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Information, prediction and uery by committee. In *Advances in Neural Information Processing Systems 1993*, pages 483–490, 1993.

[46] B. Fritzke. The LBG-U method for vector quantization - an improvement over LBG inspired from neural networks. *Neural Processing Letters*, 5(1):35–45, 1997.

[47] Bernd Fritzke. Kohonen feature maps and growing cell structures—a performance comparison. In L. Giles, S. Hanson, and J. Cowan, editors, *Advances in Neural Information Processing Systems 5*, pages 123–130. Morgan Kaufmann, San Mateo, CA, 1993.

[48] Bernd Fritzke. Growing cell structures—a self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7(9):1441–1460, 1994.

[49] K. Fukumizu. Active learning in multilayer perceptrons. In *Advances in Neural Information Processing Systems 1996*, pages 295–301, 1996.

[50] P. Grassberger and I. Procaccia. Maesuring the strangeness of strange attractors. *Physica*, 9D:189–208, 1983.

[51] P. Grassberger and I. Procaccia. Measuring the strangeness of strange attractors. *Physica*, 9D:189–208, 1983.

[52] D. Gustafson and W. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *Proc. of IEEE Conference on Decision and Control (CDC)*, pages 761–766, 1979.

[53] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[54] B. Hammer, A. Rechtien, M. Strickert, and T. Villmann. Rule extraction from self-organizing maps. In *Proc. of ICANN 2002*, pages 877–882, 2002.

[55] B. Hammer, Andreas Rechtien, M. Strickert, and Th. Villmann. Relevance learning for mental disease classification. In M. Verleysen, editor, *Proc. Of European Symposium on Artificial Neural Networks (ESANN'2005)*, pages 139–144, Brussels, Belgium, 2005. d-side publications.

[56] B. Hammer, F.-M. Schleif, and Th. Villmann. On the generalization ability of prototype-based classifiers with local relevance determination. *Technical Reports University of Clausthal IfI-05-14*, page 18 pp, 2005.

[57] B. Hammer, M. Strickert, and T. Villmann. Relevance lvq versus svm. In *Proc. of ICAISC, LNCS 3070*, pages 592–597, 2004.

[58] B. Hammer, M. Strickert, and T. Villmann. On the generalization ability of GRLVQ networks. *Neural Proc. Letters*, 21(2):109–120, 2005.

[59] B. Hammer, M. Strickert, and Th. Villmann. On the generalization ability of GRLVQ networks. *Neural Processing Letters*, 21(2):109–120, 2005.

[60] B. Hammer, M. Strickert, and Th. Villmann. Supervised neural gas with general similarity measure. *Neural Processing Letters*, 21(1):21–44, 2005.

[61] B. Hammer, T. Villmann, F.-M. Schleif, C. Albani, and W. Hermann. Learning vector quantization classification with local relevance determination for medical data. In *Proc. of ICAISC 2006*, pages 603–612, 2006.

[62] B. Hammer and Th. Villmann. Generalized relevance learning vector quantization. *Neural Networks*, 15(8-9):1059–1068, 2002.

[63] B. Hammer and Th. Villmann. Mathematical aspects of neural networks. In M. Verleysen, editor, *Proc. Of European Symposium on Artificial Neural Networks (ESANN'2003)*, pages 59–72, Brussels, Belgium, 2003. d-side.

[64] Barbara Hammer, Marc Strickert, and Thomas Villmann. Prototype based recognition of splice sites. In U. Seiffert, L.A. Jain, and P. Schweitzer, editors, *Bioinformatic using Computational Intelligence Paradigms*, pages 25–55. Springer-Verlag, 2004.

[65] M. Hasenjäger and H. Ritter. Active learning with local models. *Neural Processing Letters*, 7:107–117, 1998.

[66] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

[67] Simon Haykin. *Neural Networks - A comprehensive foundation*. Prentice Hall, New Jersey, 1999.

[68] D. Hebb. *The Organization of behaviour: a neurophysiological theory.* John Wiley & Sons, 1949.

[69] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, (2):359–266, 1989.

[70] J.-N. Hwang, J.J. Choi, S. Oh, and R. J. Marks II. Query-based learning aplied to partially trained multilayer perceptrons. *IEEE Transactions on Neural Networks*, 2:131–136, 1991.

[71] T. Jaakkola, M. Diekhans, and D. Haussler. A discrimitive framework for detecting remote protein homologies. *Journal of Computational Biology*, 7(1-2):95–114, 2000.

[72] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Compututing Surveys*, 31:264 – 323, 1999.

[73] D. Jain, A. Zongker. Feature selection: Evaluation, application and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:153–158, 1997.

[74] I.T. Jolliffe. *Principal Component Analysis*. Springer, New York, 1986.

[75] Samuel Kaski, Janne Sinkkonen, and Janne Nikkilä. Clustering gene expression data by mutual information with gene function. In Georg Dorffner, Horst Bischof, and Kurt Hornik, editors, *Artificial Neural Networks—ICANN 2001*, pages 81–86. Springer, Berlin, 2001.

[76] Samuel Kaski, Janne Sinkkonen, and Jaakko Peltonen. Bankruptcy analysis with self-organizing maps in learning metrics. *IEEE Transactions on Neural Networks*, 12:936–947, 2001.

[77] M. J. Kearns, Y. Mansur, A.Y. Ng, and D. Ron. An experimental and theoretical comparison of model selection methods. *Machine Learning*, 27:7–50, 1997.

[78] R. Ketterlinus, S-Y. Hsieh, S-H. Teng, H. Lee, and W. Pusch. Fishing for biomarkers: analyzing mass spectrometry data with the new clinprotools software. *Bio techniques*, 38(6):37–40, 2005.

[79] W. Kinzel and P. Rujan. Improving a network generalization ability by selecting examples. *Europhysics Letters*, 13:473–477, 1990.

[80] H. Klock and J.M. Buhmann. Data visualization by multidimensional scaling: A deterministic annealing approach. *Pattern recognition*, 33(4):651–699, 1999.

[81] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(12):273–324, 1997.

[82] T. Kohonen. Learning vector quantization. In *The Handbook of Brain Theory and Neural Networks*, pages 537–540. The MIT Press, Cambridge, Massachusetts, 1995.

[83] Teuvo Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Heidelberg, 1995. (2nd Ext. Ed. 1997).

[84] Teuvo Kohonen, Samuel Kaski, and Harri Lappalainen. Self-organized formation of various invariant-feature filters in the adaptive-subspace SOM. *Neural Computation*, 9:1321–1344, 1997.

[85] J. Kolen and S. Kremer. *A Field Guide to Dynamical Recurrent Networks*. IEEE Press, 2001.

[86] K.J. Lang and M.J. Witbrock. Learning to tell two spirals apart. In *Proc. of the 1988 Connectionist Models Summer School*, pages 52–59. Morgan Kaufmann, 1988.

[87] Y. Lee and C-K. Lee. Classification of multiple cancer types by multicategory support vector machines using gene expression data. *Bioinformatics*, 19(9):1132–1139, 2003.

[88] J. Li, Z. Zhang, J. Rosenzweig, Y. Wang, and D. Chan. Proteomics and bioinformatics approaches for identification of serum biomarkers to detect breast cancer. *Clinical Chemistry*, 48(8):1296–1304, 2002.

[89] L. Li, H. Tang, Z. Wu, J. Gong, M. Gruidl, J. Zhou, M. Tockman, and R.A. Clark. Data mining techniques for cancer detection using serum proteomic profiling. *Artificial Intelligence in Medicine*, 32:71–83, 2004.

[90] Daniel C. Liebler. *Introduction to Proteomics*. Humana Press, 2002.

[91] A. W-C. Liew, H. Yan, and M. Yang. Pattern recognition techniques for the emerging field of bioinformatics: A review. *Pattern Recognition*, 38:2055–2073, 2005.

[92] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28:84–95, 1980.

[93] Th. Maier, S. Klepel, U. Renner, and M. Kostrzewa. Microorganism identification and classification based on maldi-tof ms fingerprinting, 2006. Bruker Application Notes MT-80, Bruker Daltonik GmbH.

[94] T. Martinetz, S. Berkovich, and K. Schulten. Neural-gas network for vector quantization and its application to time-series prediction. *IEEE Transactions on Neural Networks*, 4(4):558–569, 1993.

[95] Thomas Martinetz and Klaus Schulten. Topology representing networks. *Neural Networks*, 7(2), 1994.

[96] Thomas M. Martinetz, Stanislav G. Berkovich, and Klaus J. Schulten. 'Neural-gas' network for vector quantization and its application to time-series prediction. *IEEE Trans. on Neural Networks*, 4(4):558–569, 1993.

[97] Ken Mauritz. MALDI-TOF Mass Spectrometry, (last visit 01.11.2006). available at: http://www.psrc.usm.edu/mauritz/images/maldi1b.jpg.

[98] J. McClelland, D. Rumelhart, and G. Hinton. *The Appeal of Parallel Distributed Processing*. MIT Press, 1987.

[99] W. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5(115), 1943.

[100] Peter Meinicke and Helge Ritter. Topographic local PCA maps. In H. Bothe and R. Rojas, editors, *Proceeding of the ICSC Symposia on Neural Computation (NC'2000) May 23-26, 2000 in Berlin, Germany*. Neuroinformatics Group, University of Bielefeld, ICSC Academic Press, 2000.

[101] IKP Stuttgart MHH Hannover and Bruker Daltonik. internal results on leukaemia, 2004.

[102] P. Mitra, C.A. Murthy, and S.K. Pal. A probabilistic active support vector learning algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3):412–418, 2004.

[103] Jeffrey S. Morris, Kevin R. Coombes, John Koomen, Keith A. Baggerly, and Ruji Kobayashi. Feature extraction and quantification for mass spectrometry in biomedical applications using the mean spectrum. *Bioinformatics*, 21(9):1764–1775, 2005.

[104] M.Strickert and B.Hammer. Merge som for temporal data. *Neurocomputing*, 64:39–72, 2005.

[105] M.Strickert, U.Seiffert, N.Sreenivasulu, W.Weschke, T.Villmann, and B.Hammer. Generalized relevance lvq (grlvq) with correlation measures for gene expression analysis. *Neurocomputing*, 69(7-9):651–659, 2006.

[106] E. Oja. Nonlinear pca: Algorithms and applications. In *Proc. Of the World Congress on Neural Networks Portland*, pages 396–400, Portland, 1993.

[107] L. Ornstein. Computer learning and the scientific method: A proposed solution to the information theoretical problem of meaning. *Journal of the Mount Sinai Hospital*, 32(4):437–494, 1965.

[108] J. Novovicova P. Paclik P. Somol, P. Pudil. Adaptive floating search methods in feature selection. *Pattern Recognition Letters*, 20:1157–1163, 1999.

[109] K. Pawelzik. *Nichtlineare Dynamik und Hirnaktivität*. Verlag Harri Deutsch, Frankfurt/M., Germany, 1991.

[110] E.F. Petricoin, A.M. Ardekani, B.A. Hitt, and P.J. Levine et al. Use of proteomic patterns in serum to identify ovarian cancer. *Lancet*, 359:572–577, 2002.

[111] J. Prados, A. Kalousis, and M. Hilario. On preprocessing of seldi-ms data and its evaluation. In *International Symposium on Computer Based Medical Systems (CBMS 2006)*, pages 953–958. IEEE press, 2006.

[112] M. Pregenzer. *Distinction Sensitive Learning Vector Quantization (DSLVQ)*. Technische Universität Graz, 1996. PhD thesis.

[113] F. Provost and T. Fawcett. Robust classification systems for imprecise environments. In AAAI Press Madison WI, editor, *Proc. of the 15th Nat. Conf. on Artif. Int.*, pages 706–707, 1998.

[114] W. Pusch, M. Flocco, S.M. Leung, H. Thiele, and M. Kostrzewa. Mass spectrometry-based clinical proteomics. *Pharmacogenomics*, 4:463–476, 2003.

[115] A. K. Qin and N. Suganthan. Growing generalized learning vector quantization with local neighborhood adaptation rule. In *Proc. of the 2th Int. Conference on Intelligent Systems 2004*, pages 524–529. IEEE, IEEE Press, 2004.

[116] L. Ralaivola, J. S. Swamidass, H. Saigo, and P. Baldi. Graph kernels for chemical informatics. *Neural Networks, special issue on Neural Networks and Kernel Methods for Structured Domains*, 18(8):1093–1110, 2005.

[117] H. Ritter, T. Martinetz, and K. Schulten. *Neural Computation and Self-Organizing Maps: An Introduction*. Addison-Wesley, M.A., 1992.

[118] D.J. Rogers and T.T. Tanimoto. A computer program for classifying plants. *Science*, 132:1115, 1960.

[119] J.-F. Rouet. Guest editorial: hypermedia and learning cognitive perspectives. *Journal of Computer Assisted Learning*, 16(2), 2000.

[120] J.W. Sammon. A non-linear mapping for data structure analysis. *IEEE Transactions in Computing*, 18:401–409, 1969.

[121] A. Sato and K. Yamada. A formulation of learning vector quantization using a new misclassification measure. In A. K. Jain, S. Venkatesh, and B. C. Lovell, editors, *Proceedings. Fourteenth International Conference on Pattern Recognition*, volume 1, pages 322–5. IEEE Computer Society, Los Alamitos, CA, USA, 1998.

[122] A. S. Sato and K. Yamada. Generalized learning vector quantization. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 423–429. MIT Press, 1995.

[123] Atsushi Sato and Kenji Yamada. An analysis of convergence in generalized LVQ. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proceedings of ICANN98, the 8th International Conference on Artificial Neural Networks*, volume 1, pages 170–176. Springer, London, 1998.

[124] A.C. Sauve and T.P. Speed. Normalization, baseline correction and alignment of high-throughput mass spectrometry data. In *Proceedings Gensips*, 2004. to be published, preprint http://stat-www.berkeley.edu/users/terry/Group/publications/Final2Gensips2004Sauve.pdf.

[125] F.-M. Schleif, U. Clauss, Th. Villmann, and B. Hammer. Supervised Relevance Neural Gas and Unified Maximum Separability Analysis for Classification of Mass Spectrometric Data. In *Proceedings of ICMLA 2004*, pages 374–379. IEEE Press, December 2004.

[126] F.-M. Schleif, T. Elssner, M. Kostrzewa, T. Villmann, and B. Hammer. Analysis and visualization of proteomic data by fuzzy labeled self organizing maps. In *Proceedings of CBMS 2006*, pages 919–924. IEEE press, 2006.

[127] F.-M. Schleif, B. Hammer, and Th. Villmann. Margin based Active Learning for LVQ Networks. In *Proc. of ESANN 2006*, pages 539–545, 2006.

[128] F.-M. Schleif, B. Hammer, and Th. Villmann. Margin based Active Learning for LVQ Networks. *NeuroComputing*, page to appear, 2006.

[129] F.-M. Schleif, T. Villmann, T. Elssner, J. Decker, and M. Kostrzewa. Machine learning and soft-computing in bioinformatics - a short journey. In *Proceedings of FLINS 2006*, pages 541–548. World Scientific Press, 2006.

[130] F.-M. Schleif, T. Villmann, and B. Hammer. Prototype based fuzzy classification in clinical proteomics. *Special issue of International Journal of Approximate Reasoning on Approximate reasoning and Machine learning for Bioinformatics*, page to appear, 2006.

[131] F.-M. Schleif, Th. Villmann, and B. Hammer. Fuzzy Labeled Soft Nearest Neighbor Classification with Relevance Learning. In *Proc. of ICMLA 2005*, pages 11–15. IEEE Press, 2005.

[132] F.-M. Schleif, Th. Villmann, and B. Hammer. Local Metric Adaptation for Soft Nearest Prototype Classification to Classify Proteomic Data. In *Fuzzy Logic and Applications: 6th Int. Workshop, WILF 2005*, Lecture Notes in Computer Science (LNCS), pages 290–296. Springer, 2006.

[133] G. Schohn and D. Cohn. Less is more: active learning with support vector machines. In *International Conference in Machine Learning*, pages 839–846, 2000.

[134] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

[135] S. Seo, M. Bode, and K. Obermayer. Soft nearest prototype classification. *IEEE Transaction on Neural Networks*, 14:390–398, 2003.

[136] S. Seo and K. Obermayer. Soft learning vector quantization. *Neural Computation*, 15:1589–1604, 2003.

[137] J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, New York, 1982.

[138] H.S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 287–294, 1992.

[139] John Shawe-Taylor. Classification accuracy based on observed margin. *Algorithmica*, 22:157–172, 1998.

[140] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

[141] Janne Sinkkonen and Samuel Kaski. Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14:217–239, 2002.

[142] P. Soille, E.J. Breen, and R. Jones. Recursive implementation of erosions and dilations along discrete lines at arbitrary angles. *IEEE T-PAMI*, 18(5):562–567, 1996.

[143] S. Sonnenburg, G. Raetsch, A. Jagota, and K. R. Mueller. New methods for splice site recognition. In *Proc. International Conference on Artificial Neural Networks, ICANN'2002, 2002*, pages 329–3360. Springer, 2002.

[144] Claudio De Stefano, Ciro D'Elia, and Angelo Marcelli. A dynamic approach to learning vector quantization. In *Proc. of the 17th Int. Conference on Pattern Recognition (ICPR'04)*, pages 601–604. IEEE, IEEE Press, 2004.

[145] M. Strickert, U. Seiffert, N. Sreenivasulu, W. Weschke, T. Villmann, and B. Hammer. Generalized relevance LVQ (GRLVQ) with correlation measures for gene expression analysis. *Neurocomputing*, 69(6–7):651–659, March 2006. ISSN: 0925-2312.

[146] M. Strickert, N. Sreenivasulu, W. Weschke, U. Seiffert, and Th. Villmann. Generalized relevance LVQ with correlation measure for biological data. In M. Verleysen, editor, *Proc. Of European Symposium on Artificial Neural Networks (ESANN'2005)*, pages 331–338, Brussels, Belgium, 2005. d-side publications.

[147] J.A. Swets and R. M. Pickett. *Evaluation of Diagnostic Systems: Methods from Signal Detection Theory*. Academic Press, New York, 1982.

[148] T.T. Tanimoto. *An Elementary Mathematical Theory of Classification and Prediction.* I.B.M. Program IBCLF, 1959.

[149] K. Tademir and E. Mernyi. Data topology visualization for the self-organizing map. In *Proc. 14th European Symposium on Artificial Neural Networks, ESANN'2006, Bruges, Belgium, 26-28 April, 2006*, pages 125–130, 2006.

[150] R. Tibshirani, T. Hastie, B. NArashimhan, S. Soltys, G. Shi, A. Kong, and Q. Le. Sample classification from protein mass spectrometry by peak probability contrasts. *Bioinformatics*, 20(17):3034–3044, 2004.

[151] S. Tong and D. Koller. Support vector machine active learning with application to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.

[152] W. S. Torgerson. Multidimensional scaling. i: Theory and method. *Psychometrica*, 17:401–419, 1952.

[153] V Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[154] V Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 2000.

[155] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to thier probabilities. *Th. of Prob. and its Appl.*, 16:264–280, 1971.

[156] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer New York, Inc., New York, NY, USA, 1995.

[157] J. Villanueva, J. Philip, D. Entenberg, and C.A. Chaparro et al. Serum peptide profiling by magnetic particle-assisted, automated sample processing and maldi-tof mass spectrometry. *Anal. Chem.*, 76:1560–1570, 2004.

[158] T. Villmann, B. Hammer, F.-M. Schleif, and T. Geweniger. Fuzzy labeled neural gas for fuzzy classification. In *Proc. of WSOM 2005*, pages 283–290, 2005.

[159] Th. Villmann. Evolutionary algorithms with subpopulations using a neural network like migration scheme and its application to real world problems. *Integrated Computer- Aided Engineering*, page to appear, 2001.

[160] Th. Villmann. Neural maps for faithful data modelling in medicine – state of the art and exemplary applications. *Neurocomputing*, 48(1–4):229–250, 2002.

[161] Th. Villmann, B. Badel, D. Kämpf, and M. Geyer. Monitoring of physiological parameters of patients and therapists during psychotherapy sessions using self-organizing maps. In H. Malmgren, M. Boga, and L. Niklasson, editors, *Artificial Neural Networks in Medicine and Biology*, pages 221–226, London, Berlin Heidelberg, 2000. Springer Verlag.

[162] Th. Villmann and H.-U. Bauer. Growing self-organizing feature maps form a hypercubical output space. In K. Lieven, editor, *Proc. Of Symposium 'Anwendung Von Fuzzy-Technologien und Neuronalen Netzen' Erfurt*, Aachen, Germany, 1995. MIT GmbH Aachen.

[163] Th. Villmann, R. Der, M. Herrmann, and Th. Martinetz. Topology Preservation in Self–Organizing Feature Maps: Exact Definition and Measurement. *IEEE Transactions on Neural Networks*, 8(2):256–266, 1997.

[164] Th. Villmann and B. Hammer. Supervised neural gas for learning vector quantization. In D. Polani, J. Kim, and T. Martinetz, editors, *Proc. of the 5th German Workshop on Artificial Life (GWAL-5)*, pages 9–16. Akademische Verlagsgesellschaft - infix - IOS Press, Berlin, 2002.

[165] Th. Villmann, B. Hammer, and F.-M. Schleif. Metrik Adaptation for Optimal Feature Classification in Learning Vector Quantization Applied to Environment Detection. In *Proc. of SOAVE 2004 - Fortschritts-Berichte VDI Reihe 10, Nr. 742*. VDI Verlag, 2004.

[166] Th. Villmann, W. Hermann, and M. Geyer. Data mining and knowledge discovery in medical applications using self-organizing maps. In R. Brause and E. Hanisch, editors, *Medical Data Analysis*, pages 138–151, Berlin, New York, Heidelberg, 2000. Lecture Notes in Computer Science 1933, Springer–Verlag.

[167] Th. Villmann, A. Hessel, and G. Plöttner. The growing SOM for estimation of the intrinsic dimension and range of psychotherapy process data. In P. P. Wang and G. M. Georgiou, editors, *Proceedings of the 3rd International Conference on Computational Intelligence and Neuroscience*, pages 72–75, Duke University and Research Triangle Park, Durham, North Carolina (USA), 1998. Association for Intelligent Machinery, Inc.

[168] Th. Villmann and E. Merényi. Extensions and modifications of SOM and its application in satellite remote sensoring processing. In H. Bothe and R.Rojas, editors, *Neural Computation 2000*, pages 765–771, Zürich, 2000. ICSC Academic Press.

[169] Th. Villmann and E. Merényi. Extensions and modifications of the Kohonen-SOM and applications in remote sensing image analysis. In U. Seiffert and L.C. Jain, editors, *Self-Organizing Maps. Recent Advances and Applications*, pages 121–145. Springer-Verlag, Heidelberg, 2001.

[170] Th. Villmann, E. Merényi, and B. Hammer. Neural maps in remote sensing image analysis. *Neural Networks*, 16(3-4):389–403, 2003.

[171] Th. Villmann, F.-M. Schleif, and B. Hammer. Supervised Neural Gas and Relevance Learning in Learning Vector Quantisation. In *Proceedings of WSOM 2003*, pages 47–52, 2003.

[172] Th. Villmann, F.-M. Schleif, and B. Hammer. Comparison of Relevance Learning Vector Quantization with other Metric Adaptive Classification Methods. *Neural Networks*, 19:610–622, 2006.

[173] Th. Villmann, F.-M. Schleif, and B. Hammer. Prototype-based fuzzy classification with local relevance for proteomics. *NeuroComputing*, 69(16-18):2425–2428, 2006.

[174] M. Wagner, D. Naik, and A. Pothen. Protocols for diesease classification from mass spectrometry data. *Proteomics*, 9(3):1692–1698, 2003.

[175] M. Wagner, D. Naik, A. Pothen, S. Kasukurti adn R. Devineni, B. Adam, O. Semmes, and G. Wright. Computational protein biomarker prediction: a case study for prostate cancer. *BMC Bioinformatics*, 5(26):open access, 2004.

[176] W. Wienholt. *Entwurf Neuronaler Netze*. Harri Deutsch, Frankfurt/M, 1996.

[177] Wikipedia. http://en.wikipedia.org/wiki/binary_classification, 2006. last visit 19.03.2006.

[178] Wikipedia. http://en.wikipedia.org/wiki/mass_spectrometry, 2006. last visit 07.10.2006.

[179] D. J. Willshaw and C. Von der Malsburg. How patterned neural connections can be set up by self–organization. *Proceedings of the Royal Society of London, Series B*, 194:431–445, 1976.

[180] J.D. Wulfkuhle, E.F. Petricoin, and L.A. Liotta. Proteomic applications for the early detection of cancer. *Nat. Rev. Cancer*, 3:267–275, 2003.

[181] E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In *Proc. of Advances in Neural Information Processing Systems 15 (NIPS)*, pages 505–512. MIT Press, 2003.

[182] Y. Yasui, M. Pepe, M. L. Thompson, A. Bao-ling, G. Wright, Y. Qu, J. Potter, M. Winget, M. Thornquist, and Z. Feng. A data-analytic strategy for protein biomarker discovery: profiling of high-dimensional proteomic data for cancer detection. *Biostatistics*, 4(3):449–454, 2003.

[183] J. S. Yu, S. Ongarello, R. Fiedler, and X.W. Chen et al. Ovarian cancer identification based on dimensionality reduction for high-throughput mass spectrometry data. *Bioinformatics*, 21(10):2200–2209, 2005.

[184] Hongtu Zhu, Chang-Yung Yu, and Heping Zhang. Tree-based disease classification using protein data. *Proteomics*, 3:1673–1677, 2003.

Index

Active learning, 83

Batch neural gas, 28
Bead based sample preparation, 14
Biomarker, 58, 98

CFE, 62, 104
Classification
  Bayes, 32
  LDA,QDA, 32
Classifier
  FSNPC, 68
  GLVQ, 45
  LVQ, 42
  SNG, 45
  Soft Nearest Prototype Classifier, 66
  Support Vector Machine, 34
  SVM, 34
Clustering
  Neural Gas, 28
Complexity measures, 74
Correlative feature elimination, 62, 104

Data analysis, 29
Datasets
  EVMS, 9
  LEUK, 9
  NCI, 9
  Proteom 1, 9
  Proteom 2, 9
  WDBC, 9
dichotomizer, 39
Dimension estimation, 29
  GPDim, 29
  Grassberger, 29
  Hausdorff dimension, 29
Dynamic LVQ, 91

Energy function
  GLVQ, 45
    penalized, 47
  GRLVQ, 51
  LGRLVQ, 55
  NPC, 66
  SNPC, 67
  SRNG, 52

Feature
  elimination, 62
  reduction, 62
  selection, 99
FSNPC, 68
Fuzzy classification, 68
Fuzzy tessellation, 70

Generalization
  Bounds
    GRLVQ, 77
    LGRLVQ, 81
Generalization bounds, 74, 81
Generalization theory, 73
GLVQ, 45
GRLVQ, 51
Growing LVQ, 91

High-dimensional data, 29

Interpolated Voronoi tessellation, 70

Label mean square error, 108
LDA, 103
Learning Vector Quantization, 42

LFSNPC, 70
Linear Discriminant Analysis, 32
LMSQE, 108
Local metric adaptation, 54, 70
local relevance profiles, 106
Localized metric, 55
LVQ, 42

Mahalanobis metric, 104
Margin based growing, 91
Metric
    local, 54
    Mahalanobis, 59
    Scaled Euclidean, 58
    Tanimoto, 61
    weighted, 58
Metric adaptation, 57

PCA, 23, 100
Penalization, 36, 47
Preparation
    Baseline correction, 18
    Normalization, 16
    Peak picking, 19
    Recalibration, 16
Principal component, 23
Principal component analysis, 23

QDA, 103
Quadratic Discriminant Analysis, 32

Receiver operator characteristics, 36
Relevance learning, 69
relevance profiles, 105
ROC, 36, 39

Sample preparation, 14
Self organizing map, 26, 99
sensitivity, 37
SNG, 45
SOM, 26, 99
specificity, 37
SRM, 82
Structural risk minimization, 82

Tanimoto similarity measure, 104

Vector Quantization, 23
Visualization, 99
Voronoi tessellation, 70